

## MacApp™ 2.0b5 Source Listings

The following page is a letter from Apple's Software Licensing department which discusses distribution of MacApp. Please note that the source code in these listings is provided for your own use—this package does not give you any rights to distribute them in any form.

The MacApp 2.0b5 Source Listings consists of 482 pages of cross-referenced source listings which were created using PasRef, an MPW tool included with the MPW Pascal product. The cross-reference includes the following source files of MacApp and MacApp building blocks (which can be found on the disks contained in the MacApp 2.0b5 product):

|     |                          |       |     |                      |
|-----|--------------------------|-------|-----|----------------------|
| 1.  | UAssociation.p           | Page  | 1   |                      |
| 2.  | UAssociation.inc1.p      | Pages | 2   | to 5                 |
| 3.  | UBusyCursor.p            | Pages | 6   | to 7                 |
| 4.  | UBusyCursor.inc1.p       | Pages | 8   | to 11                |
| 5.  | UDialog.p                | Pages | 12  | to 19                |
| 6.  | UDialog.inc1.p           | Pages | 20  | to 55                |
| 7.  | UFailure.p               | Pages | 56  | to 58                |
| 8.  | UFailure.inc1.p          | Pages | 59  | to 62                |
| 9.  | UGridView.p              | Pages | 63  | to 70                |
| 10. | UGridView.inc1.p         | Pages | 71  | to 103               |
| 11. | UList.p                  | Pages | 104 | to 107               |
| 12. | UList.inc1.p             | Pages | 108 | to 115               |
| 13. | UMacApp.p                | Pages | 116 | to 146               |
| 14. | UMacApp.Globals.p        | Pages | 147 | to 160               |
| 15. | UMacApp.TEvtHandler.p    | Pages | 161 | to 164               |
| 16. | UMacApp.TApplication.p   | Pages | 165 | to 199               |
| 17. | UMacApp.TDocument.p      | Pages | 200 | to 215               |
| 18. | UMacApp.TView.p          | Pages | 216 | to 235               |
| 19. | UMacApp.TWindow.p        | Pages | 236 | to 246               |
| 20. | UMacApp.TScroller.p      | Pages | 247 | to 253               |
| 21. | UMacApp.TControls.p      | Pages | 254 | to 268               |
| 22. | UMacApp.TCommand.p       | Pages | 269 | to 270               |
| 23. | UMacApp.TDeskScrapView.p | Pages | 271 | to 274               |
| 24. | UMacApp.TPrintHandler.p  | Pages | 275 | to 277               |
| 25. | UMAUtil.p                | Pages | 278 | to 283               |
| 26. | UMAUtil.inc1.p           | Pages | 284 | to 297               |
| 27. | UMemory.p                | Pages | 298 | to 301               |
| 28. | UMemory.inc1.p           | Pages | 302 | to 313               |
| 29. | UMenuSetup.p             | Pages | 314 | to 316               |
| 30. | UMenuSetup.inc1.p        | Pages | 317 | to 323               |
| 31. | UObject.p                | Pages | 324 | to 325               |
| 32. | UObject.inc1.p           | Page  | 326 | and pages 328 to 333 |
| 33. | ObjLib.inc1.p            | Pages | 326 | to 328               |
| 34. | UPatch.p                 | Pages | 333 | to 334               |
| 35. | UPatch.inc1.p            | Pages | 335 | to 337               |
| 36. | UPrinting.p              | Pages | 338 | to 343               |
| 37. | UPrinting.inc1.p         | Pages | 344 | to 366               |
| 38. | UTEView.p                | Pages | 367 | to 373               |

|     |                      |       |     |    |     |
|-----|----------------------|-------|-----|----|-----|
| 39. | UTEView.Globals.p    | Pages | 374 | to | 376 |
| 40. | UTEView.TTEView.p    | Pages | 377 | to | 393 |
| 41. | UTEView.TTECommand.p | Pages | 394 | to | 403 |
| 42. | UViewCoords.p        | Page  | 404 |    |     |

Each line in the source files is preceded by the number of the file it came from (using the above numbers) and the line number within that file.

The cross-reference contains an alphabetical list of all identifiers used in the source files. Following each identifier is a list of the line number with file numbers in parentheses. An asterisk (\*) following a line number indicates a definition of the variable or routine. An equal sign (=) indicates an assignment. A line number with nothing following it means a reference to an identifier.

Note that the following assembly language and MacApp debugger source files were not included in the listing (interested users can list them from the files on the MacApp 2.0b5 disks):

- CommonObjLib.a
- Macros.a
- UBusyCursor.a
- UFailure.a
- UInspector.p
- UInspector.inc1.p
- UMAUtil.a
- UObject.a
- UTrace.a
- UTrace.p
- UTrace.inc1.p
- UViewCoords.a
- UWriteInWindow.p
- UWriteInWindow.inc1.p

Also not included in this listing are the MacApp resource files, make files, and sample programs.





Dear MacApp User:

MacApp™ is a copyrighted program owned by Apple Computer, Inc. and is designed for combination with Macintosh™ programs written by you. However, this copy of the MacApp Source Listings does not give you any rights to redistribute MacApp in any form.

If you wish to distribute programs which include object code from the MacApp library or building blocks you will need to complete a MacApp Object Code Distribution Agreement and return it with the appropriate annual licensing fee to Apple's Software Licensing Department. A copy of that agreement is included in the MacApp package; it is also available by writing to:

Software Licensing Department  
Mailstop 28B  
Apple Computer, Inc.  
20525 Mariani Avenue  
Cupertino, CA 95014

The annual fee is \$100.00 if any of your applications are distributed commercially (i.e. in an effort to earn a profit) or \$10.00 if all of your applications are distributed non-commercially. The Software Licensing Department does provide commercial licensees with upgrades to new versions of software, but no updates will be provided to holders of the non-commercial distribution license.

Once you have returned the Agreement and fee, you may distribute any number of different applications for one year, subject to the terms and conditions of the Agreement. The only requirement is that you send a list of any additional applications to Apple's Software Licensing Department within 90 days of when you first ship the applications. This may be sent via mail or to SW.LICENSE on AppleLink™.

Apple will not support or answer questions from end-users of your applications, so you should provide user support for applications you build using MacApp. Problems with the MacApp software should be reported to Apple through a designated contact at your company or institution.

If you have any questions, please contact us at (408) 973-4667.

Sincerely,

Software Licensing

Apple Computer, Inc.  
20525 Mariani Avenue  
Cupertino, California 95014  
408 996-1010  
TLX 171-576



```

1 1 --      {SP}
1 2 --      {UAssociation.p}
1 3 --      {Copyright © 1988 by Apple Computer Inc. All rights reserved.}
1 4 --
1 5 --      UNIT UAssociation;
1 6 --
1 7 --      INTERFACE
1 8 --
1 9 --      {-----+
1 10 --      | Uses
1 11 --      +-----}
1 12 --      USES
1 13 --          {$LOAD MacIntf.LOAD}
1 14 --          MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
1 15 --          {$LOAD}
1 16 --          UMAUtil,
1 17 --          UFailure,
1 18 --          UObject,
1 19 --          UList;
1 20 --
1 21 --      {-----+
1 22 --      | Types
1 23 --      +-----}
1 24 --      TYPE
1 25 --      {-----+
1 26 --      | TEntry
1 27 --      +-----}
1 28 --      TEntry = OBJECT(TObject)
1 29 --
1 30 --          fKey:      StringHandle;
1 31 --          fValue:     StringHandle;
1 32 --
1 33 --          PROCEDURE TEntry.IEntry (itsKey, itsValue: Str255);
1 34 --
1 35 --          PROCEDURE TEntry.Free; OVERRIDE;
1 36 --
1 37 --      {$IFC qDebug}
1 38 --          PROCEDURE TEntry.Fields (PROCEDURE DoToField (fieldName: Str255;
1 39 --                                  fieldAddr: Ptr;
1 40 --                                  fieldType: INTEGER)); OVERRIDE;
1 41 --
1 42 --      {$ENDC}
1 43 --
1 44 --          END (TEntry);
1 45 --
1 46 --      {-----+
1 47 --      | TEntriesList
1 48 --      +-----}
1 49 --      TEntriesList = OBJECT(TSortedList)
1 50 --
1 51 --          FUNCTION TEntriesList.Compare (item1, item2: TObject): INTEGER; OVERRIDE;
1 52 --
1 53 --          END (TEntriesList);
1 54 --
1 55 --
1 56 --      {-----+
1 57 --      | TAssociation
1 58 --      +-----}
1 59 --      TAssociation = OBJECT(TObject)
1 60 --
1 61 --          fEntries:      TEntriesList;
1 62 --
1 63 --          PROCEDURE TAssociation.IAssociation;
1 64 --
1 65 --          PROCEDURE TAssociation.Free; OVERRIDE;
1 66 --
1 67 --          FUNCTION TAssociation.ValueAt (keyStr: Str255; VAR valueStr: Str255): BOOLEAN;
1 68 --
1 69 --          FUNCTION TAssociation.KeyAt (valueStr: Str255; VAR keyStr: Str255): BOOLEAN;
1 70 --
1 71 --          PROCEDURE TAssociation.EachEntryDo (PROCEDURE DoToEntry (theEntry: TEntry));
1 72 --
1 73 --          FUNCTION TAssociation.EntryWithKey (keyStr: Str255): TEntry;
1 74 --
1 75 --          FUNCTION TAssociation.EntryWithValue (valueStr: Str255): TEntry;
1 76 --
1 77 --          FUNCTION TAssociation.FirstEntryThat (FUNCTION TestEntry (theEntry: TEntry): BOOLEAN): TEntry;
1 78 --
1 79 --          PROCEDURE TAssociation.InsertEntry (keyStr, valueStr: Str255);
1 80 --
1 81 --          PROCEDURE TAssociation.RemoveValueAt (keyStr: Str255);
1 82 --
1 83 --          PROCEDURE TAssociation.RemoveKeyAt (valueStr: Str255);
1 84 --
1 85 --      {$IFC qDebug}
1 86 --          PROCEDURE TAssociation.Fields (PROCEDURE DoToField (fieldName: Str255;
1 87 --                                  fieldAddr: Ptr;
1 88 --                                  fieldType: INTEGER)); OVERRIDE;
1 89 --
1 90 --      {$ENDC}
1 91 --
1 92 --          END (TAssociation);
1 93 --
1 94 --      IMPLEMENTATION
1 95 --
1 96 --      {$I UAssociation.incl.p}

```

```

2 1 --      {$P}
2 2 --      {UAssociation.incl.p}
2 3 --      {Copyright © 1988 by Apple Computer Inc. All rights reserved.}
2 4 --
2 5 --      {$S MAMain}      ( Everything goes in here)
2 6 --
2 7 --      (*****
2 8 --      (*      T E n t r y      *)
2 9 --      (*****
2 10 --      {-----+
2 11 --      |      IEntry      |
2 12 --      +-----}
2 13 -- A  PROCEDURE TEntry.IEntry (itsKey, itsValue: Str255);
2 14 --
2 15 --      VAR
2 16 --          fi:      FailInfo;
2 17 --
2 18 --      {-----+
2 19 --      |      HdliEntry      |
2 20 --      +-----}
2 21 -- B  PROCEDURE HdliEntry (error: OSErr; message: LONGINT);
2 22 --
2 23 0- B  BEGIN
2 24 --      DisposIfHandle(fKey);
2 25 --      Free;
2 26 -0 B  END;
2 27 --
2 28 0- A  BEGIN
2 29 --      CatchFailures(fi, HdliEntry);
2 30 --
2 31 --      fKey := NewString(itsKey);
2 32 --      FailNIL(fKey);
2 33 --
2 34 --      fValue := NewString(itsValue);
2 35 --      FailNIL(fValue);
2 36 --
2 37 --      Success(fi);
2 38 -0 A  END {TEntry.IEntry};
2 39 --
2 40 --      {-----+
2 41 --      |      Free      |
2 42 --      +-----}
2 43 -- A  PROCEDURE TEntry.Free; OVERRIDE;
2 44 --
2 45 0- A  BEGIN
2 46 --      DisposIfHandle(fKey);
2 47 --      DisposIfHandle(fValue);
2 48 --      INHERITED Free;
2 49 -0 A  END {TEntry.Free};
2 50 --
2 51 --      {$IFC qDebug}
2 52 --      {-----+
2 53 --      |      Fields      |
2 54 --      +-----}
2 55 -- A  PROCEDURE TEntry.Fields (PROCEDURE DoToField (fieldName: Str255;
2 56 --                                fieldAddr: Ptr;
2 57 --                                fieldType: INTEGER)); OVERRIDE;
2 58 0- A  BEGIN
2 59 --      DoToField('TEntry', NIL, bClass);
2 60 --      DoToField('fKey', @fKey, bStringHandle);
2 61 --      DoToField('fValue', @fValue, bStringHandle);
2 62 --      INHERITED Fields(DoToField);
2 63 -0 A  END {TEntry.Fields};
2 64 --      {$ENDC}
2 65 --
2 66 --
2 67 --      (*****
2 68 --      (*      T E n t r y      *)
2 69 --      (*****
2 70 --      {-----+
2 71 --      |      Compare      |
2 72 --      +-----}
2 73 -- A  FUNCTION TEntriesList.Compare (item1, item2: TObj): INTEGER;
2 74 --
2 75 0- A  BEGIN
2 76 --      IF TEntry(item1).fKey^^ < TEntry(item2).fKey^^ THEN
2 77 --          Compare := kALessThanB
2 78 --      ELSE IF TEntry(item1).fKey^^ > TEntry(item2).fKey^^ THEN
2 79 --          Compare := kAGreaterThanB
2 80 --      ELSE
2 81 --          Compare := 0;
2 82 -0 A  END {TEntriesList.Compare};
2 83 --
2 84 --
2 85 --      (*****
2 86 --      (*      T A s s o c i a t i o n      *)
2 87 --      (*****
2 88 --      {-----+
2 89 --      |      IAssociation      |
2 90 --      +-----}
2 91 -- A  PROCEDURE TAssociation.IAssociation;
2 92 --
2 93 --      VAR
2 94 --          anEntriesList: TEntriesList;
2 95 --
2 96 0- A  BEGIN
2 97 --      New(anEntriesList);
2 98 --      FailNIL(anEntriesList);

```

```

2  99 --      anEntriesList.IList;
2 100 --      fEntries := anEntriesList;
2 101 --      {SIFC qDebug}
2 102 --      fEntries.SetEltType('TEntry');
2 103 --      {$ENDC}
2 104 -0 A  END {TAssociation.IAssociation};
2 105 --
2 106 --      {-----+
2 107 --      |   Free                               |
2 108 --      +-----+
2 109 -0 A  PROCEDURE TAssociation.Free; OVERRIDE;
2 110 --
2 111 --      {-----+
2 112 --      |   FreeEntry                           |
2 113 --      +-----+
2 114 -0 B  PROCEDURE FreeEntry (theEntry: TEntry);
2 115 --
2 116 0- B  BEGIN
2 117 --      theEntry.Free;
2 118 -0 B  END;
2 119 --
2 120 0- A  BEGIN
2 121 --      EachEntryDo(FreeEntry);
2 122 --      fEntries.Free;
2 123 --      INHERITED Free;
2 124 -0 A  END {TAssociation.Free};
2 125 --
2 126 --      {-----+
2 127 --      |   EachEntryDo                         |
2 128 --      +-----+
2 129 -0 A  PROCEDURE TAssociation.EachEntryDo (PROCEDURE DoToEntry (theEntry: TEntry));
2 130 --
2 131 0- A  BEGIN
2 132 --      fEntries.Each(DoToEntry);
2 133 -0 A  END {TAssociation.EachEntryDo};
2 134 --
2 135 --
2 136 --      {-----+
2 137 --      |   EntryWithKey                       |
2 138 --      +-----+
2 139 -0 A  FUNCTION TAssociation.EntryWithKey (keyStr: Str255): TEntry;
2 140 --
2 141 --      {-----+
2 142 --      |   TestKey                             |
2 143 --      +-----+
2 144 -0 B  FUNCTION TestKey (theEntry: TObject): INTEGER;
2 145 --
2 146 0- B  BEGIN
2 147 --      IF keyStr < TEntry(theEntry).fKey^^ THEN
2 148 --          TestKey := kALessThanB
2 149 --      ELSE IF keyStr > TEntry(theEntry).fKey^^ THEN
2 150 --          TestKey := kAGreaterThanB
2 151 --      ELSE
2 152 --          TestKey := kAEqualB;
2 153 -0 B  END;
2 154 --
2 155 0- A  BEGIN
2 156 --      EntryWithKey := TEntry(fEntries.Search(TestKey));
2 157 -0 A  END;
2 158 --
2 159 --      {-----+
2 160 --      |   EntryWithValue                     |
2 161 --      +-----+
2 162 -0 A  FUNCTION TAssociation.EntryWithValue (valueStr: Str255): TEntry;
2 163 --
2 164 --      {-----+
2 165 --      |   MatchesValue                       |
2 166 --      +-----+
2 167 -0 B  FUNCTION MatchesValue (theEntry: TEntry): BOOLEAN;
2 168 --
2 169 0- B  BEGIN
2 170 --      MatchesValue := valueStr = theEntry.fValue^^;
2 171 -0 B  END;
2 172 --
2 173 0- A  BEGIN
2 174 --      EntryWithValue := FirstEntryThat(MatchesValue);
2 175 -0 A  END;
2 176 --
2 177 --
2 178 --      {-----+
2 179 --      |   FirstEntryThat                     |
2 180 --      +-----+
2 181 -0 A  FUNCTION TAssociation.FirstEntryThat (FUNCTION TestEntry (theEntry: TEntry): BOOLEAN): TEntry;
2 182 --
2 183 0- A  BEGIN
2 184 --      FirstEntryThat := TEntry(fEntries.FirstThat(TestEntry));
2 185 -0 A  END {TAssociation.FirstEntryThat};
2 186 --
2 187 --      {-----+
2 188 --      |   InsertEntry                         |
2 189 --      +-----+
2 190 -0 A  PROCEDURE TAssociation.InsertEntry (keyStr, valueStr: Str255);
2 191 --
2 192 --      VAR
2 193 --          anEntry:      TEntry;
2 194 --          fi:           FailInfo;
2 195 --
2 196 --      {-----+
2 197 --      |   Hd1SetString                       |

```

```

2 198 -- +-----+
2 199 -- B   PROCEDURE HdlSetString (error: OSErr; message: LONGINT);
2 200 --
2 201 0- B   BEGIN
2 202 --       {If we couldn't increase the handle size, set it to something reasonable}
2 203 --       anEntry.fValue^^ := '';
2 204 0- B   END;
2 205 --
2 206 -- +-----+
2 207 -- |   TestKey                               |
2 208 -- +-----+
2 209 -- B   FUNCTION TestKey (theEntry: TEntry): BOOLEAN;
2 210 --
2 211 0- B   BEGIN
2 212 --       TestKey := theEntry.fKey^^ = keyStr;
2 213 0- B   END;
2 214 --
2 215 0- A   BEGIN
2 216 --       anEntry := EntryWithKey(keyStr);
2 217 --       IF anEntry <> NIL THEN
2 218 1-         BEGIN
2 219 --             CatchFailures(fi, HdlSetString);
2 220 --             SetString(anEntry.fValue, valueStr);
2 221 --             FailMemError;                               { SetString may increase the handle size }
2 222 --             Success(fi);
2 223 1-         END
2 224 --       ELSE
2 225 1-         BEGIN
2 226 --             New(anEntry);
2 227 --             FailNIL(anEntry);
2 228 --             anEntry.IEntry(keyStr, valueStr);
2 229 --             fEntries.Insert(anEntry);
2 230 1-         END;
2 231 0- A   END {TAssociation.InsertEntry};
2 232 --
2 233 -- +-----+
2 234 -- |   KeyAt                               |
2 235 -- +-----+
2 236 -- A   FUNCTION TAssociation.KeyAt (valueStr: Str255; VAR keyStr: Str255): BOOLEAN;
2 237 --
2 238 --       VAR
2 239 --         theEntry:      TEntry;
2 240 --
2 241 --       +-----+
2 242 --       |   MatchesValue                     |
2 243 --       +-----+
2 244 -- B   FUNCTION MatchesValue (theEntry: TEntry): BOOLEAN;
2 245 --
2 246 0- B   BEGIN
2 247 --       MatchesValue := valueStr = theEntry.fValue^^;
2 248 0- B   END;
2 249 --
2 250 0- A   BEGIN
2 251 --       theEntry := FirstEntryThat(MatchesValue);
2 252 --       IF theEntry = NIL THEN
2 253 1-         BEGIN
2 254 --             keyStr := '';
2 255 --             KeyAt := FALSE;
2 256 1-         END
2 257 --       ELSE
2 258 1-         BEGIN
2 259 --             keyStr := theEntry.fKey^^;
2 260 --             KeyAt := TRUE;
2 261 1-         END;
2 262 0- A   END {TAssociation.KeyAt};
2 263 --
2 264 -- +-----+
2 265 -- |   RemoveValueAt                         |
2 266 -- +-----+
2 267 -- A   PROCEDURE TAssociation.RemoveValueAt (keyStr: Str255);
2 268 --
2 269 --       VAR
2 270 --         theEntry:      TEntry;
2 271 --
2 272 --       +-----+
2 273 --       |   TestKey                               |
2 274 --       +-----+
2 275 -- B   FUNCTION TestKey (anItem: TObjct): INTEGER;
2 276 --
2 277 0- B   BEGIN
2 278 --       IF keyStr < TEntry(anItem).fKey^^ THEN
2 279 --         TestKey := -1
2 280 --       ELSE IF keyStr > TEntry(anItem).fKey^^ THEN
2 281 --         TestKey := 1
2 282 --       ELSE
2 283 --         TestKey := 0;
2 284 0- B   END;
2 285 --
2 286 0- A   BEGIN
2 287 --       theEntry := TEntry(fEntries.Search(TestKey));
2 288 --       IF theEntry <> NIL THEN
2 289 --         fEntries.Delete(theEntry);
2 290 0- A   END {TAssociation.RemoveValueAt};
2 291 --
2 292 -- +-----+
2 293 -- |   RemoveKeyAt                             |
2 294 -- +-----+
2 295 -- A   PROCEDURE TAssociation.RemoveKeyAt (valueStr: Str255);
2 296 --

```

```

2 297 -- VAR
2 298 --     theEntry:      TEntry;
2 299 --
2 300 --     {-----+
2 301 --     | MatchesValue |
2 302 --     +-----}
2 303 -- B FUNCTION MatchesValue (theEntry: TEntry): BOOLEAN;
2 304 --
2 305 0- B BEGIN
2 306 --     MatchesValue := valueStr = theEntry.fValue^^;
2 307 -0 B END;
2 308 --
2 309 0- A BEGIN
2 310 --     theEntry := FirstEntryThat(MatchesValue);
2 311 --     IF theEntry <> NIL THEN
2 312 --         fEntries.Delete(theEntry);
2 313 -0 A END {TAssociation.RemoveKeyAt};
2 314 --
2 315 --     {-----+
2 316 --     | ValueAt |
2 317 --     +-----}
2 318 -- A FUNCTION TAssociation.ValueAt (keyStr: Str255; VAR valueStr: Str255): BOOLEAN;
2 319 --
2 320 -- VAR
2 321 --     theEntry:      TEntry;
2 322 --
2 323 --     {-----+
2 324 --     | CompareKey |
2 325 --     +-----}
2 326 -- B FUNCTION CompareKey (theEntry: TObject): INTEGER;
2 327 --
2 328 0- B BEGIN
2 329 --     IF keyStr < TEntry(theEntry).fKey^^ THEN
2 330 --         CompareKey := -1
2 331 --     ELSE IF keyStr > TEntry(theEntry).fKey^^ THEN
2 332 --         CompareKey := 1
2 333 --     ELSE
2 334 --         CompareKey := 0;
2 335 -0 B END;
2 336 --
2 337 0- A BEGIN
2 338 --     theEntry := TEntry(fEntries.Search(CompareKey));
2 339 --     IF theEntry = NIL THEN
2 340 1- BEGIN
2 341 --         valueStr := '';
2 342 --         ValueAt := FALSE;
2 343 -1 END
2 344 --     ELSE
2 345 1- BEGIN
2 346 --         valueStr := theEntry.fValue^^;
2 347 --         ValueAt := TRUE;
2 348 -1 END;
2 349 -0 A END {TAssociation.ValueAt};
2 350 --
2 351 --     {$IFC qDebug}
2 352 --     {-----+
2 353 --     | Fields |
2 354 --     +-----}
2 355 -- A PROCEDURE TAssociation.Fields (PROCEDURE DoToField (fieldName: Str255;
2 356 --     fieldAddr: Ptr;
2 357 --     fieldType: INTEGER)); OVERRIDE;
2 358 --
2 359 0- A BEGIN
2 360 --     DoToField('TAssociation', NIL, bClass);
2 361 --     DoToField('fEntries', @fEntries, bObject);
2 362 --     INHERITED Fields(DoToField);
2 363 -0 A END {TAssociation.Fields};
2 364 --     {$ENDC}
2 365 --
1 97 --
1 98 -- END.

```

```

3 1 --  { $P }
3 2 --  { UBusyCursor.p }
3 3 --  { Copyright © 1985 - 1988 Apple Computer, Inc. All rights reserved. }
3 4 --
3 5 --  { Conditional compilation flags used by this unit:
3 6 --
3 7 --      qNeedsROM128K    True skips code for Mac XL, false leaves it in.
3 8 --
3 9 --      qTrace           True surrounds trap patches with $D+ and $D++ (because
3 10 --                    the default is $D++), false skips these directives.
3 11 --
3 12 --      The settings of $D, $R and $N are imported from the UMAUtil interface.
3 13 --  }
3 14 --
3 15 --  UNIT UBusyCursor:
3 16 --
3 17 --      {
3 18 --          THEORY OF OPERATION
3 19 --
3 20 --      The UBusyCursor unit implements a mechanism for automatically setting
3 21 --      the cursor to the watch when the application is "busy."
3 22 --
3 23 --      An application is considered "busy" if it hasn't called GetNextEvent
3 24 --      or EventAvail for a given number of ticks, where 60 ticks equals one
3 25 --      second. The default time is defined by kWatchDelay, which is 120 ticks
3 26 --      or 2 seconds--it can be changed by calling BusyDelay. After this period
3 27 --      of time has elapsed with no call to GetNextEvent or EventAvail, the
3 28 --      cursor is changed to the watch. On the next call to GetNextEvent or
3 29 --      EventAvail, the cursor is restored to its state before it was changed to
3 30 --      the watch.
3 31 --
3 32 --      Changing the cursor to the watch is done by the VBL task AWatchTask.
3 33 --      AWatchTask's execution frequency is set to kWatchDelay, or by calling
3 34 --      BusyDelay. When AWatchTask is executed it sets the cursor to the
3 35 --      watch and resets AWatchTask's vblCount.
3 36 --
3 37 --      The traps InitCursor, SetCursor and SetCCursor are patched so that
3 38 --      they "remember" the cursor being set before executing the trap. This
3 39 --      will allow us to restore the cursor after it has been changed to the
3 40 --      busy watch.
3 41 --
3 42 --      The EventAvail and GetNextEvent traps are patched such that, before
3 43 --      executing the trap, AWatchTask's vblCount is reset, and if the busy
3 44 --      cursor is displayed, then the cursor is restored to the last cursor
3 45 --      set by SetCursor or SetCCursor.
3 46 --
3 47 --      Note that this unit supports the Macintosh XL, though the XL implement-
3 48 --      ation differs slightly from that described above. See the source code.
3 49 --  }
3 50 --
3 51 --  INTERFACE
3 52 --
3 53 --  {-----+
3 54 --  | Uses
3 55 --  +-----+
3 56 --  USES
3 57 --      { $LOAD MacIntf.LOAD }
3 58 --      MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
3 59 --      { $LOAD }
3 60 --      UMAUtil,
3 61 --      UPatch,
3 62 --      UFailure:
3 63 --
3 64 --  {-----+
3 65 --  | Constants
3 66 --  +-----+
3 67 --  CONST
3 68 --      kWatchDelay      =      2*60;    (default # of 1/60 sec. ticks before
3 69 --                                     cursor changes to a watch)
3 70 --
3 71 --  {-----+
3 72 --  | Global Declarations
3 73 --  +-----+
3 74 --  PROCEDURE BusyActivate(entering: BOOLEAN);
3 75 --      { Call BusyActivate if you want to activate or deactivate the busy cursor
3 76 --        mechanism. This is call by MacApp when losing/gaining control to a
3 77 --        desk accessory or switcher partition. }
3 78 --
3 79 --  PROCEDURE BusyDelay(newDelay: INTEGER; forceBusy: BOOLEAN);
3 80 --      { Call BusyDelay if you want to change the busy cursor delay.
3 81 --        newDelay should be in 1/60 seconds; a value <= 0 means don't change the
3 82 --        delay. If forceBusy is TRUE, then the watch is put up immediately,
3 83 --        otherwise it doesn't go up until the required time has passed.
3 84 --        (BusyDelay respects the state flags in the cursor info record (ie.,
3 85 --        changeToWatch and inControl.) }
3 86 --
3 87 --  PROCEDURE ResetBusyCursor;
3 88 --      { Call ResetBusyCursor if you want to change the cursor back to an arrow
3 89 --        and reset the time before changing back to a watch. This is not
3 90 --        usually called directly by the application. Instead, it
3 91 --        is called each time GetNextEvent and EventAvail is called, by
3 92 --        patches installed by BusyInstall. }
3 93 --
3 94 --  PROCEDURE BusyInstall;
3 95 --      { BusyInstall installs the busy cursor mechanism. Typically this is
3 96 --        called once during program initialization. It installs the
3 97 --        VBL task and patches the traps GetNextEvent, EventAvail, InitCursor,
3 98 --        SetCursor and SetCCursor. }

```



```
3 99 --  
3 100 -- PROCEDURE BusyRemove:  
3 101 --   { BusyRemove uninstalls the busy cursor mechanism. Typically this is  
3 102 --     called once during program termination. It removes the VBL task  
3 103 --     and unpatches the patched traps. }  
3 104 --  
3 105 -- IMPLEMENTATION  
3 106 --  
3 107 -- ({I UBusyCursor.incl.p})
```

```

4 1 -- {SP}
4 2 -- { UBusyCursor.incl.p }
4 3 -- { Copyright 1985-1988 Apple Computer, Inc. All rights reserved. }
4 4 --
4 5 -- {-----+
4 6 -- | Local types |
4 7 -- +-----}
4 8 -- TYPE
4 9 --     CursorInfo = RECORD
4 10 --         inColor:    BOOLEAN;    { Is the saved cursor in color? }
4 11 --         origCursor: Cursor;    { Cursor at the time the busy cursor was
4 12 --                               put up, if not in color }
4 13 --         origCCursor: CCrsrHandle; { Cursor at the time the busy cursor was
4 14 --                               put up, if in color }
4 15 --         watchDelay: INTEGER;    { time in 1/60 second before cursor
4 16 --                               changes to watch }
4 17 --         inControl:   BOOLEAN;    { managed by MacApp; TRUE iff MacApp is
4 18 --                               in control; if FALSE we don't
4 19 --                               change the cursor at all }
4 20 --         changeToWatch: BOOLEAN;  { if TRUE, we automatically switch to
4 21 --                               the watch in the VBL task and switch
4 22 --                               to origCursor on a call to
4 23 --                               GetNextEvent or EventAvail;
4 24 --                               applications can changed this as
4 25 --                               necessary }
4 26 --         watchOn:     BOOLEAN;    { TRUE if the busy cursor on }
4 27 --         spare:       BOOLEAN;    { reserved for future use }
4 28 --         watchCursor: Cursor;    { the watch cursor }
4 29 --         q:           QElem;      { vbl queue element for changing the
4 30 --                               cursor }
4 31 --     END;
4 32 --
4 33 -- {-----+
4 34 -- | Local variables |
4 35 -- +-----}
4 36 -- VAR
4 37 --     gCursorInfo:    CursorInfo;
4 38 --     gEAPatch:       TrapPatch;    { patch for EventAvail }
4 39 --     gNEPatch:       TrapPatch;    { patch for GetNextEvent }
4 40 --     gICPatch:       TrapPatch;    { patch for InitCursor }
4 41 --     gSCPatch:       TrapPatch;    { patch for SetCursor }
4 42 --     gSCCPatch:      TrapPatch;    { patch for SetCCursor }
4 43 --
4 44 --
4 45 -- {-----+
4 46 -- | External declarations |
4 47 -- +-----}
4 48 -- A PROCEDURE AWatchTask; EXTERNAL; { busy cursor VBL task }
4 49 --
4 50 -- { $IFC NOT qNeedsROM128K }
4 51 -- A PROCEDURE XLBusyDelay(millisec: LONGINT); EXTERNAL;
4 52 -- A PROCEDURE XLBusyImage(c: Cursor); EXTERNAL;
4 53 -- A PROCEDURE XLCursorInit; EXTERNAL;
4 54 -- { $ENDC }
4 55 --
4 56 --
4 57 -- {-----+
4 58 -- | Forward declarations |
4 59 -- +-----}
4 60 -- A PROCEDURE BusyReset(delayTicks: INTEGER); FORWARD;
4 61 -- A PROCEDURE SetMacAppCursor (VAR theCursor: Cursor); FORWARD;
4 62 -- A PROCEDURE SetCMacAppCursor (theCCursor: CCrsrHandle); FORWARD;
4 63 -- A PROCEDURE InitMacAppCursor; FORWARD;
4 64 --
4 65 --
4 66 -- { $MAMain }
4 67 -- {-----+
4 68 -- | BusyActivate |
4 69 -- +-----}
4 70 -- A PROCEDURE BusyActivate(entering: BOOLEAN);
4 71 --
4 72 -- 0- A BEGIN
4 73 --     BusyReset(gCursorInfo.watchDelay);
4 74 --     gCursorInfo.inControl := entering;
4 75 --
4 76 -- { $IFC NOT qNeedsROM128K }
4 77 --     IF gConfiguration.machineType = envXL THEN
4 78 --         IF NOT entering THEN
4 79 --             XLCursorInit;
4 80 --         { $ENDC }
4 81 -- 0 A END { BusyActivate };
4 82 --
4 83 --
4 84 -- { $MAMain }
4 85 -- {-----+
4 86 -- | BusyDelay |
4 87 -- +-----}
4 88 -- A PROCEDURE BusyDelay(newDelay: INTEGER; forceBusy: BOOLEAN);
4 89 --
4 90 -- 0- A BEGIN
4 91 --     WITH gCursorInfo DO
4 92 --         BEGIN
4 93 --             IF newDelay > 0 THEN { save new delay time }
4 94 --                 watchDelay := newDelay;
4 95 --
4 96 --             IF forceBusy THEN
4 97 --                 newDelay := 1; { trigger on next tick }
4 98 --

```

```

4 99 --          IF newDelay > 0 THEN          { reset timer }
4 100 --          BusyReset(newDelay);
4 101 --          END;
4 102 --0 A  END { BusyDelay };
4 103 --
4 104 --
4 105 --      {$S MAINit}
4 106 --      {-----+
4 107 --      |  BusyInstall      |
4 108 --      +-----+}
4 109 -- A  PROCEDURE BusyInstall;
4 110 --
4 111 --0 A  BEGIN
4 112 --      UserROMMap(TRUE);
4 113 --
4 114 --      { Setup the gCursorInfo record }
4 115 --
4 116 --      gCursorInfo.watchCursor := GetCursor(watchCursor)^^;
4 117 --      WITH gCursorInfo DO
4 118 --          BEGIN
4 119 --              inControl := TRUE; { we are in control during initialization }
4 120 --              changeToWatch := TRUE;
4 121 --              watchOn := FALSE;
4 122 --              watchDelay := kWatchDelay;
4 123 --              inColor := FALSE;
4 124 --              origCursor := arrow;
4 125 --
4 126 --      {$IFC NOT qNeedsROM128K}
4 127 --          IF gConfiguration.machineType = envXL THEN
4 128 --              XLBusyImage(gCursorInfo.watchCursor)
4 129 --          ELSE
4 130 --              ($ENDC)
4 131 --          BEGIN
4 132 --
4 133 --              { Setup the VBL task }
4 134 --              WITH q.vblQElem DO
4 135 --                  BEGIN
4 136 --                      qType := ORD(vType);
4 137 --                      vblAddr := @AWatchTask;
4 138 --                      vblCount := kWatchDelay;
4 139 --                      vblPhase := 0;
4 140 --                  END;
4 141 --
4 142 --              { Patch the necessary traps for non-XL operation }
4 143 --              FailOSErr(TailPatch(gSCPatch, $A851, @SetMacappCursor)); { SetCursor }
4 144 --              FailOSErr(TailPatch(gICPatch, $A850, @InitMacappCursor)); { InitCursor }
4 145 --              IF gConfiguration.hasColorQD THEN
4 146 --                  FailOSErr(TailPatch(gSCCPatch, $AA1C, @SetMacappCursor)); { SetCCursor }
4 147 --
4 148 --              { Install the VBL task }
4 149 --              FailOSErr(VInstall(@q));
4 150 --          END;
4 151 --      END;
4 152 --
4 153 --      { Patch the traps applicable to Mac and XL operation }
4 154 --      FailOSErr(TailPatch(gGNEPatch, $A970, @ResetBusyCursor)); { GetNextEvent }
4 155 --      FailOSErr(TailPatch(gEAPatch, $A971, @ResetBusyCursor)); { EventAvail }
4 156 --
4 157 --      { Turn on busy cursor right now; init watchDelay }
4 158 --      BusyDelay(kWatchDelay, TRUE);
4 159 --0 A  END { BusyInstall };
4 160 --
4 161 --
4 162 --      {$S MATerminate}
4 163 --      {-----+
4 164 --      |  BusyRemove      |
4 165 --      +-----+}
4 166 -- A  PROCEDURE BusyRemove;
4 167 --
4 168 --      { Undo changes for automatic busy cursor--removes the VBL task and
4 169 --      unpatches the patched traps supporting the busy cursor. }
4 170 --
4 171 --      VAR
4 172 --          e: OSErr;
4 173 --0 A  BEGIN
4 174 --      {$IFC NOT qNeedsROM128K}
4 175 --          IF gConfiguration.machineType = envXL THEN
4 176 --              BEGIN
4 177 --                  XLCursorInit;
4 178 --              END
4 179 --          ELSE
4 180 --              ($ENDC)
4 181 --          BEGIN
4 182 --              { Remove the VBL task }
4 183 --              e := VRemove(@gCursorInfo.q);
4 184 --
4 185 --              { Unpatch the non-XL patches }
4 186 --              UnpatchTrap(gICPatch);
4 187 --              UnpatchTrap(gSCPatch);
4 188 --              UnpatchTrap(gSCCPatch);
4 189 --          END;
4 190 --
4 191 --      { These patches apply to all Macs }
4 192 --      UnpatchTrap(gGNEPatch);
4 193 --      UnpatchTrap(gEAPatch);
4 194 --0 A  END { BusyRemove };
4 195 --
4 196 --
4 197 --      {$S MAMain}

```

```

4 198 -- {-----+
4 199 -- |   BusyReset                               |
4 200 -- +-----+
4 201 -- A  PROCEDURE BusyReset(delayTicks: INTEGER);
4 202 --
4 203 -- A  BEGIN
4 204 --         WITH gCursorInfo DO
4 205 --             IF inControl THEN { ??? should we make this test ??? }
4 206 --                 IF changeToWatch THEN
4 207 --                     ($IFC NOT qNeedsROM128K)
4 208 --                         IF gConfiguration.machineType = envXL THEN
4 209 --                             XLBusyDelay((Ord4(delayTicks) * 1000) DIV 60)
4 210 --                         ELSE
4 211 --                             ($ENDC)
4 212 --                     BEGIN
4 213 --                         IF watchOn THEN
4 214 --                             IF inColor THEN
4 215 --                                 SetCCursor(origCCursor)
4 216 --                             ELSE
4 217 --                                 SetCursor(origCursor);
4 218 --                                 q.vblQElem.vblCount := delayTicks;
4 219 --                             END;
4 220 -- A  END { BusyReset };
4 221 --
4 222 --
4 223 -- ($S Main)
4 224 -- ($IFC qTrace){$D+}{$ENDC}
4 225 -- {-----+
4 226 -- |   BusyTurnOff                               |
4 227 -- +-----+
4 228 -- A  PROCEDURE BusyTurnOff;
4 229 --
4 230 --         { This is called from InitMacAppCursor, SetMacAppCursor and SetCMacAppCursor.
4 231 --           It sets gCursorInfo fields to indicate that the cursor is not the
4 232 --           busy watch. }
4 233 --
4 234 -- A  BEGIN
4 235 --         WITH gCursorInfo DO
4 236 --             IF inControl THEN
4 237 --                 IF changeToWatch THEN
4 238 --                     BEGIN
4 239 --                         watchOn := FALSE; { anyone that sets the busy cursor should set this TRUE explicitly }
4 240 --                         q.vblQElem.vblCount := gCursorInfo.watchDelay;
4 241 --                     END;
4 242 -- A  END { BusyTurnOff };
4 243 -- ($IFC qTrace){$D+}{$ENDC}
4 244 --
4 245 --
4 246 -- ($S Main)
4 247 -- ($IFC qTrace){$D+}{$ENDC}
4 248 -- {-----+
4 249 -- |   InitMacAppCursor                         |
4 250 -- +-----+
4 251 -- A  PROCEDURE InitMacAppCursor;
4 252 --
4 253 --         { Called when the InitCursor trap is executed. After completion, we jump
4 254 --           to the ROM InitCursor. }
4 255 --
4 256 -- A  BEGIN
4 257 --         SetMacAppCursor(arrow);
4 258 -- A  END { InitMacAppCursor };
4 259 -- ($IFC qTrace){$D+}{$ENDC}
4 260 --
4 261 --
4 262 -- ($IFC qTrace){$D+}{$ENDC}
4 263 -- ($S MAMain)
4 264 -- {-----+
4 265 -- |   ResetBusyCursor                         |
4 266 -- +-----+
4 267 -- A  PROCEDURE ResetBusyCursor;
4 268 --
4 269 --         { Patches GetNextEvent and EventAvail to reset the busy cursor.
4 270 --           Also callable from a program. }
4 271 --
4 272 -- A  BEGIN
4 273 --         SetupA5;           { ***** Called from trap patches ***** }
4 274 --         BusyReset(gCursorInfo.watchDelay);
4 275 --         RestoreA5;
4 276 -- A  END { ResetBusyCursor };
4 277 -- ($IFC qTrace){$D+}{$ENDC}
4 278 --
4 279 --
4 280 -- ($S Main)
4 281 -- ($IFC qTrace){$D+}{$ENDC}
4 282 -- {-----+
4 283 -- |   SetCMacAppCursor                       |
4 284 -- +-----+
4 285 -- A  PROCEDURE SetCMacAppCursor (theCCursor: CCrsrcHandle);
4 286 --
4 287 --         { The SetCCursor patch, used to remember the color cursor being set.
4 288 --           Installed as a "tail" patch, meaning the original SetCCursor trap
4 289 --           is called after this code has completed.}
4 290 --
4 291 -- A  BEGIN
4 292 --         SetupA5;           { ***** Called from trap patches ***** }
4 293 --         BusyTurnOff;
4 294 --         WITH gCursorInfo DO
4 295 --             BEGIN
4 296 --                 inColor := TRUE;

```

```

4 297 --      { Save a copy of the color cursor }
4 298 --      gCursorInfo.origCursor := theCursor;
4 299 -1      END;
4 300 --
4 301 --      RestoreA5;
4 302 -0 A  END { SetMacAppCursor };
4 303 --      {$IFC qTrace}{$D+}{$ENDC}
4 304 --
4 305 --
4 306 --      {$S Main}      { must be in Main segment, and cannot call to any other segment
4 307 --                     because SetCursor is called from the AWatchTask VBL task }
4 308 --      {$IFC qTrace}{$D+}{$ENDC}
4 309 --      {-----+
4 310 --      |   SetMacAppCursor
4 311 --      +-----}
4 312 - A  PROCEDURE SetMacAppCursor (VAR theCursor: Cursor);
4 313 --
4 314 --      { Patches SetCursor to remember the cursor being set.  Installed as a
4 315 --        "tail" patch, meaning the original SetCursor trap is called after
4 316 --        this code has completed.  Also called from InitMacAppCursor. }
4 317 --
4 318 0- A  BEGIN
4 319 --      SetupA5;      { ***** Called from trap patches *****}
4 320 --      BusyTurnOff;
4 321 --      { If we are setting the cursor to the busy watch, then don't save it }
4 322 --      IF @theCursor <> @gCursorInfo.watchCursor THEN
4 323 1-          BEGIN
4 324 --          gCursorInfo.inColor := FALSE;
4 325 --          gCursorInfo.origCursor := theCursor;
4 326 -1          END
4 327 --      ELSE
4 328 --          gCursorInfo.watchOn := TRUE;      { because BusyTurnOff set it to FALSE }
4 329 --      RestoreA5;
4 330 -0 A  END { SetMacAppCursor };
4 331 --      {$IFC qTrace}{$D+}{$ENDC}
3 108 --
3 109 --      END {UBusyCursor}.

```

```

5 1 --      {$P}
5 2 --      {UDialog.p}
5 3 --      {Copyright © 1988 by Apple Computer Inc. All rights reserved.}
5 4 --
5 5 --      UNIT UDialog;
5 6 --
5 7 --      INTERFACE
5 8 --
5 9 --      {-----+
5 10 --      | Uses
5 11 --      +-----}
5 12 --      USES
5 13 --          ($LOAD MacIntf.LOAD)
5 14 --          MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
5 15 --          ($LOAD UMacApp.LOAD)
5 16 --          UMAUtil, UViewCoords, UFailure,
5 17 --          UMemory, UMenuSetup, UObject, UList, UAssociation, UMacApp,
5 18 --          ($LOAD)
5 19 --          UTEView,
5 20 --          Sane,
5 21 --          Script;
5 22 --
5 23 --      {-----+
5 24 --      | Constants
5 25 --      +-----}
5 26 --      CONST
5 27 --
5 28 --          { Miscellaneous Constants}
5 29 --
5 30 --          kPreferColor      = TRUE;           { argument to IIcon
5 31 --          kControlOn       = TRUE;           { argument to ICheckBox and IRadio
5 32 --
5 33 --          kFrame            = [adnLineTop, adnLineLeft, adnLineBottom, adnLineRight];
5 34 --
5 35 --          Chr00              = CHR(0);         { Needed for ControlCharSet
5 36 --          Chr1F              = CHR($1F);
5 37 --
5 38 --          kMaxTEWidth        = $2B0;          { Maximum width for TEditText autoscrolling
5 39 --
5 40 --      {-----+
5 41 --      | Types
5 42 --      +-----}
5 43 --      TYPE
5 44 --
5 45 --          ControlCharSet = SET OF Chr00..Chr1F;
5 46 --
5 47 --      {-----+
5 48 --      | TDialogView
5 49 --      +-----}
5 50 --      DialogViewTemplatePtr = ^DialogViewTemplate;
5 51 --      DialogViewTemplate = RECORD
5 52 --          defaultItem: IDType;
5 53 --          cancelItem: IDType;
5 54 --      END;
5 55 --
5 56 --      TDialogView = OBJECT(TView)
5 57 --          fDefaultItem: IDType; { The identifier of the default item
5 58 --          fCancelItem: IDType;  { The identifier of item that cancels dialog
5 59 --          fParamTxt: TAssociation; { the list of TEntries for replacement
5 60 --          fCurrentEditText: TEditText; { Currently selected edit text item
5 61 --          fTEView: TDialogTEView; { Used for editing in EditText views
5 62 --          fDismissed: BOOLEAN; { Has the dialog been dismissed?
5 63 --          fDismitter: IDType; { ID of view that caused dismissal
5 64 --
5 65 --      {$IFC qProceduralViews}
5 66 --          PROCEDURE TDialogView.IDialogView (itsDocument: TDocument; itsSuperView: TView;
5 67 --          itsLocation, itsSize: VPoint;
5 68 --          itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 69 --          itsDefItemID, itsCancelItemID: IDType);
5 70 --      {$ENDC}
5 71 --
5 72 --      {$IFC qTemplateViews}
5 73 --          PROCEDURE TDialogView.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
5 74 --      {$ENDC}
5 75 --
5 76 --      {$IFC qWriteTemplates}
5 77 --          PROCEDURE TDialogView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 78 --
5 79 --          PROCEDURE TDialogView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 80 --      {$ENDC}
5 81 --
5 82 --          PROCEDURE TDialogView.Free; OVERRIDE;
5 83 --
5 84 --          FUNCTION TDialogView.CanDismiss (dismissing: IDType): BOOLEAN;
5 85 --
5 86 --          PROCEDURE TDialogView.CantDeselect (theEditText: TEditText);
5 87 --
5 88 --          PROCEDURE TDialogView.Close; OVERRIDE;
5 89 --
5 90 --          PROCEDURE TDialogView.DismissDialog (dismitter: IDType);
5 91 --
5 92 --          FUNCTION TDialogView.DoCommandKey(ch: CHAR; VAR info: EventInfo): TCommand; OVERRIDE;
5 93 --          { Called when a keyDown event is received and the command key is down. }
5 94 --
5 95 --          FUNCTION TDialogView.DoKeyCommand (ch: Char; aKeyCode: INTEGER;
5 96 --          VAR info: EventInfo): TCommand; OVERRIDE;
5 97 --
5 98 --          PROCEDURE TDialogView.DoSelectEditText (theEditText: TEditText; selectChars: BOOLEAN);

```

```

5  99 --
5 100 --      PROCEDURE TDialogView.EachEditText (PROCEDURE DoToEditText (theEditText: TEditText));
5 101 --
5 102 --      FUNCTION TDialogView.GetDialogView: TView; OVERRIDE;
5 103 --
5 104 --      FUNCTION TDialogView.HandleCR (callingView: TView; VAR info: EventInfo): TCommand;
5 105 --
5 106 --      FUNCTION TDialogView.HandleTab (callingView: TView; VAR info: EventInfo): TCommand;
5 107 --
5 108 --      PROCEDURE TDialogView.DoChoice (origView: TView; itsChoice: INTEGER); OVERRIDE;
5 109 --
5 110 --      FUNCTION TDialogView.MakeTEView: TDialogTEView;
5 111 --
5 112 --      PROCEDURE TDialogView.ParamTxt (keyStr, valueStr: Str255);
5 113 --
5 114 --      FUNCTION TDialogView.PoseModally: IDType;
5 115 --
5 116 --      PROCEDURE TDialogView.ReplaceText (theText: Handle);
5 117 --
5 118 --      PROCEDURE TDialogView.SelectEditText (itsIdentifier: IDType; selectChars: BOOLEAN);
5 119 --
5 120 --      PROCEDURE TDialogView.SurveyEditText (VAR first, last, next, previous: TEditText);
5 121 --
5 122 --      ($IFC qDebug)
5 123 --          PROCEDURE TDialogView.Fields (PROCEDURE DoToField (fieldName: Str255;
5 124 --              fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 125 --      ($ENDC)
5 126 --
5 127 --          END (TDialogView);
5 128 --
5 129 --
5 130 --      {-----+
5 131 --      | TButton |
5 132 --      +-----+
5 133 --      ButtonTemplatePtr = ^ButtonTemplate;
5 134 --      ButtonTemplate = RECORD
5 135 --          itsLabel: Str255;      { Actually, variable length }
5 136 --      END;
5 137 --
5 138 --      TButton = OBJECT (TctlMgr)
5 139 --
5 140 --      ($IFC qProceduralViews)
5 141 --          PROCEDURE TButton.IButton (itsSuperView: TView; itsLocation, itsSize: VPoint;
5 142 --              itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 143 --              itsLabel: Str255);
5 144 --      ($ENDC)
5 145 --
5 146 --      ($IFC qTemplateViews)
5 147 --          PROCEDURE TButton.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
5 148 --      ($ENDC)
5 149 --
5 150 --      ($IFC qWriteTemplates)
5 151 --          PROCEDURE TButton.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 152 --
5 153 --          PROCEDURE TButton.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 154 --      ($ENDC)
5 155 --
5 156 --          END (TButton);
5 157 --
5 158 --
5 159 --      {-----+
5 160 --      | TCheckBox |
5 161 --      +-----+
5 162 --      CheckBoxTemplatePtr = ^CheckBoxTemplate;
5 163 --      CheckBoxTemplate = RECORD
5 164 --          isOn: BOOLEAN;
5 165 --          itsLabel: Str255;      { Actually, variable length }
5 166 --      END;
5 167 --
5 168 --      TCheckBox = OBJECT (TctlMgr)
5 169 --
5 170 --      ($IFC qProceduralViews)
5 171 --          PROCEDURE TCheckBox.ICheckBox (itsSuperView: TView; itsLocation, itsSize: VPoint;
5 172 --              itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 173 --              itsLabel: Str255; isTurnedOn: BOOLEAN);
5 174 --      ($ENDC)
5 175 --
5 176 --      ($IFC qTemplateViews)
5 177 --          PROCEDURE TCheckBox.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
5 178 --      ($ENDC)
5 179 --
5 180 --      ($IFC qWriteTemplates)
5 181 --          PROCEDURE TCheckBox.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 182 --
5 183 --          PROCEDURE TCheckBox.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 184 --      ($ENDC)
5 185 --
5 186 --          PROCEDURE TCheckBox.DoChoice (origView: TView; itsChoice: INTEGER); OVERRIDE;
5 187 --
5 188 --          FUNCTION TCheckBox.IsOn: BOOLEAN;
5 189 --
5 190 --          PROCEDURE TCheckBox.SetState (state, redraw: BOOLEAN);
5 191 --
5 192 --          PROCEDURE TCheckBox.Toggle (redraw: BOOLEAN);
5 193 --
5 194 --          PROCEDURE TCheckBox.ToggleIf (matchState, redraw: BOOLEAN);
5 195 --
5 196 --          END (TCheckBox);
5 197 --

```

```

5 198 --
5 199 -- {-----+
5 200 -- | TRadio |
5 201 -- +-----}
5 202 -- RadioTemplatePtr = ^RadioTemplate;
5 203 -- RadioTemplate = RECORD
5 204 --     isOn:          BOOLEAN;
5 205 --     itsLabel:      Str255;      { Actually, variable length }
5 206 --     END;
5 207 --
5 208 -- TRadio = OBJECT (TctlMgr)
5 209 --
5 210 -- {SIFC qProceduralViews}
5 211 --     PROCEDURE TRadio.IRadio (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 212 --                             itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 213 --                             itsLabel: Str255; isTurnedOn: BOOLEAN);
5 214 -- {SENDC}
5 215 --
5 216 -- {SIFC qTemplateViews}
5 217 --     PROCEDURE TRadio.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 218 -- {SENDC}
5 219 --
5 220 -- {SIFC qWriteTemplates}
5 221 --     PROCEDURE TRadio.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 222 --
5 223 --     PROCEDURE TRadio.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 224 -- {SENDC}
5 225 --
5 226 --     PROCEDURE TRadio.DoChoice (origView: TView; itsChoice: INTEGER); OVERRIDE;
5 227 --
5 228 --     FUNCTION TRadio.IsOn: BOOLEAN;
5 229 --
5 230 --     PROCEDURE TRadio.SetState (state, redraw: BOOLEAN);
5 231 --
5 232 --     PROCEDURE TRadio.Toggle (redraw: BOOLEAN);
5 233 --
5 234 --     PROCEDURE TRadio.ToggleIf (matchState, redraw: BOOLEAN);
5 235 --
5 236 --     END (TRadio);
5 237 --
5 238 --
5 239 -- {-----+
5 240 -- | TCluster |
5 241 -- +-----}
5 242 -- ClusterTemplatePtr = ^ClusterTemplate;
5 243 -- ClusterTemplate = RECORD
5 244 --     itsLabel:      Str255;      { Actually, variable length }
5 245 --     END;
5 246 --
5 247 -- TCluster = OBJECT (TControl)
5 248 --
5 249 --     fRsrcID:        INTEGER;
5 250 --     fIndex:         INTEGER;
5 251 --     fDataHandle:    StringHandle;
5 252 --
5 253 -- {SIFC qProceduralViews}
5 254 --     PROCEDURE TCluster.ICluster (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 255 --                                 itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 256 --                                 itsRsrcID, itsIndex: INTEGER);
5 257 -- {SENDC}
5 258 --
5 259 -- {SIFC qTemplateViews}
5 260 --     PROCEDURE TCluster.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 261 -- {SENDC}
5 262 --
5 263 -- {SIFC qWriteTemplates}
5 264 --     PROCEDURE TCluster.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 265 --
5 266 --     PROCEDURE TCluster.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 267 -- {SENDC}
5 268 --
5 269 --     PROCEDURE TCluster.Free; OVERRIDE;
5 270 --
5 271 --     PROCEDURE TCluster.Draw (area: Rect); OVERRIDE;
5 272 --
5 273 --     PROCEDURE TCluster.DoChoice (origView: TView; itsChoice: INTEGER); OVERRIDE;
5 274 --
5 275 --     PROCEDURE TCluster.GetLabel (VAR theLabel: Str255);
5 276 --
5 277 --     PROCEDURE TCluster.ReleaseLabel;
5 278 --
5 279 --     FUNCTION TCluster.ReportCurrent: IDType;
5 280 --
5 281 --     PROCEDURE TCluster.SetLabel(theLabel: Str255; redraw: BOOLEAN);
5 282 --
5 283 -- {SIFC qDebug}
5 284 --     PROCEDURE TCluster.Fields (PROCEDURE DoToField (fieldName: Str255;
5 285 --                                                     fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 286 -- {SENDC}
5 287 --     END (TCluster);
5 288 --
5 289 --
5 290 -- {-----+
5 291 -- | TIcon |
5 292 -- +-----}
5 293 -- IconTemplatePtr = ^IconTemplate;
5 294 -- IconTemplate = RECORD
5 295 --     preferColor:    BOOLEAN;
5 296 --     rsrcID:         INTEGER;

```



```

5 297 --          END;
5 298 --
5 299 --      TIcon =          OBJECT (TControl)
5 300 --
5 301 --          fPreferColor:  BOOLEAN;
5 302 --          fRsrcID:       INTEGER;
5 303 --          fDataHandle:   Handle;
5 304 --
5 305 --      {$IFC qProceduralViews}
5 306 --          PROCEDURE TIcon.IIcon (itsSuperView: TView; itsLocation, itsSize: VPoint;
5 307 --                                itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 308 --                                itsRsrcID: INTEGER; preferColor: BOOLEAN);
5 309 --      {$ENDC}
5 310 --
5 311 --      {$IFC qTemplateViews}
5 312 --          PROCEDURE TIcon.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
5 313 --      {$ENDC}
5 314 --
5 315 --      {$IFC qWriteTemplates}
5 316 --          PROCEDURE TIcon.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 317 --
5 318 --          PROCEDURE TIcon.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 319 --      {$ENDC}
5 320 --
5 321 --          PROCEDURE TIcon.Free; OVERRIDE;
5 322 --
5 323 --          PROCEDURE TIcon.Draw (area: Rect); OVERRIDE;
5 324 --
5 325 --          PROCEDURE TIcon.ReleaseIcon;
5 326 --
5 327 --          PROCEDURE TIcon.SetIcon (theIcon: Handle; redraw: BOOLEAN);
5 328 --
5 329 --      {$IFC qDebug}
5 330 --          PROCEDURE TIcon.Fields (PROCEDURE DoToField (fieldName: Str255;
5 331 --                                fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 332 --      {$ENDC}
5 333 --          END {TIcon};
5 334 --
5 335 --
5 336 --      {-----+
5 337 --      | TPattern                                     |
5 338 --      +-----}
5 339 --      PatternTemplatePtr = ^PatternTemplate;
5 340 --      PatternTemplate = RECORD
5 341 --          preferColor:  BOOLEAN;
5 342 --          rsrcID:       INTEGER;
5 343 --      END;
5 344 --
5 345 --      TPattern =          OBJECT (TControl)
5 346 --
5 347 --          fPreferColor:  BOOLEAN;
5 348 --          fRsrcID:       INTEGER;
5 349 --          fDataHandle:   Handle;
5 350 --
5 351 --      {$IFC qProceduralViews}
5 352 --          PROCEDURE TPattern.IPattern (itsSuperView: TView; itsLocation, itsSize: VPoint;
5 353 --                                       itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 354 --                                       itsRsrcID: INTEGER; preferColor: BOOLEAN);
5 355 --      {$ENDC}
5 356 --
5 357 --      {$IFC qTemplateViews}
5 358 --          PROCEDURE TPattern.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
5 359 --      {$ENDC}
5 360 --
5 361 --      {$IFC qWriteTemplates}
5 362 --          PROCEDURE TPattern.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 363 --
5 364 --          PROCEDURE TPattern.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 365 --      {$ENDC}
5 366 --
5 367 --          PROCEDURE TPattern.Free; OVERRIDE;
5 368 --
5 369 --          PROCEDURE TPattern.Draw (area: Rect); OVERRIDE;
5 370 --
5 371 --          PROCEDURE TPattern.ReleasePattern;
5 372 --
5 373 --          PROCEDURE TPattern.SetPattern (thePattern: Handle; redraw: BOOLEAN);
5 374 --
5 375 --      {$IFC qDebug}
5 376 --          PROCEDURE TPattern.Fields (PROCEDURE DoToField (fieldName: Str255;
5 377 --                                fieldAddr: Ptr;
5 378 --                                fieldType: INTEGER)); OVERRIDE;
5 379 --      {$ENDC}
5 380 --          END {TPattern};
5 381 --
5 382 --
5 383 --      {-----+
5 384 --      | TPicture                                     |
5 385 --      +-----}
5 386 --      PictureTemplatePtr = ^PictureTemplate;
5 387 --      PictureTemplate = RECORD
5 388 --          rsrcID:       INTEGER;
5 389 --      END;
5 390 --
5 391 --      TPicture =          OBJECT (TControl)
5 392 --
5 393 --          fRsrcID:       INTEGER;
5 394 --          fDataHandle:   PicHandle;
5 395 --

```

```

5 396 --      ($IFC qProceduralViews)
5 397 --          PROCEDURE TPicture.IPicture (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 398 --                                     itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 399 --                                     itsRsrcID: INTEGER);
5 400 --      ($ENDC)
5 401 --
5 402 --      ($IFC qTemplateViews)
5 403 --          PROCEDURE TPicture.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 404 --      ($ENDC)
5 405 --
5 406 --      ($IFC qWriteTemplates)
5 407 --          PROCEDURE TPicture.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 408 --
5 409 --          PROCEDURE TPicture.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 410 --      ($ENDC)
5 411 --
5 412 --          PROCEDURE TPicture.Free; OVERRIDE;
5 413 --
5 414 --          PROCEDURE TPicture.Draw (area: Rect); OVERRIDE;
5 415 --
5 416 --          PROCEDURE TPicture.ReleasePicture;
5 417 --
5 418 --          PROCEDURE TPicture.SetPicture (thePicture: PicHandle; redraw: BOOLEAN);
5 419 --
5 420 --      ($IFC qDebug)
5 421 --          PROCEDURE TPicture.Fields (PROCEDURE DoToField (fieldName: Str255;
5 422 --                                     fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 423 --      ($ENDC)
5 424 --          END (TPicture);
5 425 --
5 426 --
5 427 --      {-----+
5 428 --      | TPopup                                     |
5 429 --      +-----}
5 430 --      PopupTemplatePtr = ^PopupTemplate;
5 431 --      PopupTemplate = RECORD
5 432 --          rsrcID:          INTEGER;
5 433 --          currentItem:    INTEGER;
5 434 --          itemOffset:     INTEGER;
5 435 --      END;
5 436 --
5 437 --      TPopup =          OBJECT (TControl)
5 438 --
5 439 --          fRsrcID:      INTEGER;
5 440 --          fMenuID:      INTEGER;
5 441 --          fMenuHandle:  MenuHandle;
5 442 --          fCurrentItem: INTEGER;
5 443 --          fItemOffset:  INTEGER;
5 444 --
5 445 --      ($IFC qProceduralViews)
5 446 --          PROCEDURE TPopup.IPopup (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 447 --                                   itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 448 --                                   itsRsrcID, itsCurrentItem, itsItemOffset: INTEGER);
5 449 --      ($ENDC)
5 450 --
5 451 --      ($IFC qTemplateViews)
5 452 --          PROCEDURE TPopup.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 453 --      ($ENDC)
5 454 --
5 455 --      ($IFC qWriteTemplates)
5 456 --          PROCEDURE TPopup.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 457 --
5 458 --          PROCEDURE TPopup.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 459 --      ($ENDC)
5 460 --
5 461 --          PROCEDURE TPopup.Free; OVERRIDE;
5 462 --
5 463 --          PROCEDURE TPopup.AdjustBotRight;
5 464 --
5 465 --          PROCEDURE TPopup.CalcLabelRect (VAR theRect: Rect);
5 466 --
5 467 --          PROCEDURE TPopup.CalcMenuRect (VAR theRect: Rect);
5 468 --
5 469 --          FUNCTION TPopup.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
5 470 --                                          VAR hysteresis: Point): TCommand; OVERRIDE;
5 471 --
5 472 --          PROCEDURE TPopup.Draw (area: Rect); OVERRIDE;
5 473 --
5 474 --          PROCEDURE TPopup.DrawLabel (area: Rect);
5 475 --
5 476 --          PROCEDURE TPopup.DrawPopupBox (area: Rect);
5 477 --
5 478 --          FUNCTION TPopup.GetCurrentItem: INTEGER;
5 479 --
5 480 --          PROCEDURE TPopup.GetItemText (item: INTEGER; VAR theText: Str255);
5 481 --
5 482 --          PROCEDURE TPopup.ReleasePopup;
5 483 --
5 484 --          PROCEDURE TPopup.SetCurrentItem (item: INTEGER; redraw: BOOLEAN);
5 485 --
5 486 --          PROCEDURE TPopup.SetPopup (theMenu: MenuHandle; theRsrcID, currentItem: INTEGER;
5 487 --                                     redraw: BOOLEAN);
5 488 --
5 489 --      ($IFC qDebug)
5 490 --          PROCEDURE TPopup.Fields (PROCEDURE DoToField (fieldName: Str255;
5 491 --                                     fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 492 --      ($ENDC)
5 493 --          END (TPopup);
5 494 --

```

```

5 495 --
5 496 --      {-----+
5 497 --      | TStaticText |
5 498 --      +-----+
5 499 --      StaticTextTemplatePtr = ^StaticTextTemplate;
5 500 --      StaticTextTemplate = RECORD
5 501 --          just:    INTEGER;
5 502 --          data:    Str255;    { Actually, variable length }
5 503 --      END;
5 504 --
5 505 --      TStaticText = OBJECT (TControl)
5 506 --
5 507 --          fRsrcID:    INTEGER;
5 508 --          fIndex:    INTEGER;
5 509 --          fJust:    INTEGER;
5 510 --          fDataHandle: Handle;
5 511 --
5 512 --      ($IFC qProceduralViews)
5 513 --          PROCEDURE TStaticText.IStaticText (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 514 --              itsHSizeDet, itsVSizeDet: SizeDeterminer;
5 515 --              itsRsrcID, itsIndex: INTEGER);
5 516 --      ($ENDC)
5 517 --
5 518 --      ($IFC qTemplateViews)
5 519 --          PROCEDURE TStaticText.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 520 --      ($ENDC)
5 521 --
5 522 --      ($IFC qWriteTemplates)
5 523 --          PROCEDURE TStaticText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 524 --
5 525 --          PROCEDURE TStaticText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 526 --      ($ENDC)
5 527 --
5 528 --          PROCEDURE TStaticText.Free; OVERRIDE;
5 529 --
5 530 --          FUNCTION TStaticText.DoSubstitution (VAR realText: Handle): BOOLEAN;
5 531 --
5 532 --          PROCEDURE TStaticText.Draw (area: Rect); OVERRIDE;
5 533 --
5 534 --          PROCEDURE TStaticText.GetText (VAR theText: Str255);
5 535 --
5 536 --          PROCEDURE TStaticText.ImageText (text: Ptr; length: LONGINT; box: Rect;
5 537 --              just: INTEGER);
5 538 --
5 539 --          PROCEDURE TStaticText.ReleaseText;
5 540 --
5 541 --          PROCEDURE TStaticText.SetJustification (theJust: INTEGER; redraw: BOOLEAN);
5 542 --
5 543 --          PROCEDURE TStaticText.SetText (theText: Str255; redraw: BOOLEAN);
5 544 --
5 545 --      ($IFC qDebug)
5 546 --          PROCEDURE TStaticText.Fields (PROCEDURE DoToField (fieldName: Str255;
5 547 --              fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 548 --      ($ENDC)
5 549 --          END (TStaticText);
5 550 --
5 551 --
5 552 --      {-----+
5 553 --      | TEditText |
5 554 --      +-----+
5 555 --      EditTextTemplatePtr = ^EditTextTemplate;
5 556 --      EditTextTemplate = RECORD
5 557 --          maxChars:    INTEGER;
5 558 --          controlChars: ControlCharSet;
5 559 --      END;
5 560 --
5 561 --      TEditText = OBJECT (TStaticText)
5 562 --
5 563 --          fControlChars: ControlCharSet;
5 564 --          fMaxChars:    INTEGER;
5 565 --          fTextView:    TTextView;
5 566 --
5 567 --      ($IFC qProceduralViews)
5 568 --          PROCEDURE TEditText.IEditText (itsSuperview: TView; itsLocation, itsSize: VPoint;
5 569 --              itsMaxChars: INTEGER);
5 570 --      ($ENDC)
5 571 --
5 572 --      ($IFC qTemplateViews)
5 573 --          PROCEDURE TEditText.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
5 574 --      ($ENDC)
5 575 --
5 576 --      ($IFC qWriteTemplates)
5 577 --          PROCEDURE TEditText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 578 --
5 579 --          PROCEDURE TEditText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
5 580 --      ($ENDC)
5 581 --
5 582 --          PROCEDURE TEditText.DoIdle (phase: IdlePhase); OVERRIDE;
5 583 --
5 584 --          FUNCTION TEditText.DoKeyCommand (ch: CHAR; aKeyCode: INTEGER; VAR info: EventInfo): TCommand; OVERRIDE;
5 585 --
5 586 --          FUNCTION TEditText.DoMenuCommand (aCmdNumber: INTEGER): TCommand; OVERRIDE;
5 587 --
5 588 --          FUNCTION TEditText.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
5 589 --              VAR hysteresis: Point): TCommand; OVERRIDE;
5 590 --
5 591 --          PROCEDURE TEditText.DoSetupMenus; OVERRIDE;
5 592 --
5 593 --          FUNCTION TEditText.DoSubstitution (VAR realText: Handle): BOOLEAN; OVERRIDE;

```

```

5 594 --
5 595 --      PROCEDURE TEditText.Draw (area: Rect): OVERRIDE;
5 596 --
5 597 --      PROCEDURE TEditText.GetText (VAR theText: Str255): OVERRIDE;
5 598 --
5 599 --      PROCEDURE TEditText.ImageText (text: Ptr; length: LONGINT; box: Rect;
5 600 --                                   just: INTEGER): OVERRIDE;
5 601 --
5 602 --      PROCEDURE TEditText.InstallSelection (wasActive, beActive: BOOLEAN): OVERRIDE;
5 603 --
5 604 --      PROCEDURE TEditText.Locate (h, v: VCoordinate; invalidate: BOOLEAN): OVERRIDE;
5 605 --
5 606 --      PROCEDURE TEditText.Resize (width, height: VCoordinate; invalidate: BOOLEAN): OVERRIDE;
5 607 --
5 608 --      PROCEDURE TEditText.RestartEdit;
5 609 --
5 610 --      PROCEDURE TEditText.SetJustification (theJust: INTEGER; redraw: BOOLEAN): OVERRIDE;
5 611 --
5 612 --      PROCEDURE TEditText.SetSelection (selStart, selEnd: INTEGER; redraw: BOOLEAN);
5 613 --
5 614 --      PROCEDURE TEditText.SetText (theText: Str255; redraw: BOOLEAN): OVERRIDE;
5 615 --
5 616 --      PROCEDURE TEditText.StartEdit (selectChars: BOOLEAN; theTEView: TDialogTEView);
5 617 --
5 618 --      FUNCTION TEditText.StopEdit: BOOLEAN;
5 619 --
5 620 --      FUNCTION TEditText.Validate: BOOLEAN; OVERRIDE;
5 621 --
5 622 --      ($IFC qDebug)
5 623 --      PROCEDURE TEditText.Fields (PROCEDURE DoToField (fieldName: Str255;
5 624 --                                   fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 625 --      ($ENDC)
5 626 --      END (TEditText);
5 627 --
5 628 --      {-----+
5 629 --      | TNumberText |
5 630 --      +-----}
5 631 --      NumberTextTemplatePtr = ^NumberTextTemplate;
5 632 --      NumberTextTemplate = RECORD
5 633 --          value: LONGINT;
5 634 --          minimum: LONGINT;
5 635 --          maximum: LONGINT;
5 636 --      END;
5 637 --
5 638 --      TNumberText = OBJECT (TEditText)
5 639 --
5 640 --          fMinimum: LONGINT;
5 641 --          fMaximum: LONGINT;
5 642 --
5 643 --      ($IFC qProceduralViews)
5 644 --      PROCEDURE TNumberText.INumberText (itsSuperView: TView; itsLocation, itsSize: VPoint;
5 645 --                                           itsValue, itsMinimum, itsMaximum: LONGINT);
5 646 --      ($ENDC)
5 647 --
5 648 --      ($IFC qTemplateViews)
5 649 --      PROCEDURE TNumberText.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr): OVERRIDE;
5 650 --      ($ENDC)
5 651 --
5 652 --      ($IFC qWriteTemplates)
5 653 --      PROCEDURE TNumberText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr): OVERRIDE;
5 654 --
5 655 --      PROCEDURE TNumberText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr): OVERRIDE;
5 656 --      ($ENDC)
5 657 --
5 658 --      FUNCTION TNumberText.GetValue: LONGINT;
5 659 --
5 660 --      PROCEDURE TNumberText.SetValue (newValue: LONGINT; redraw: BOOLEAN);
5 661 --
5 662 --      FUNCTION TNumberText.Validate: BOOLEAN; OVERRIDE;
5 663 --
5 664 --      ($IFC qDebug)
5 665 --      PROCEDURE TNumberText.Fields (PROCEDURE DoToField (fieldName: Str255;
5 666 --                                   fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 667 --      ($ENDC)
5 668 --      END (TNumberText);
5 669 --
5 670 --      {-----+
5 671 --      | TDialogTEView |
5 672 --      +-----}
5 673 --      TDialogTEView = OBJECT (TTEView)
5 674 --
5 675 --          fEditText: TEditText;
5 676 --
5 677 --      PROCEDURE TDialogTEView.DrawContents; OVERRIDE;
5 678 --
5 679 --      PROCEDURE TDialogTEView.InstallEditText (theEditText: TEditText; selectChars: BOOLEAN);
5 680 --
5 681 --      PROCEDURE TDialogTEView.InstallSelection (wasActive, beActive: BOOLEAN); OVERRIDE;
5 682 --
5 683 --      PROCEDURE TDialogTEView.RecalcText; OVERRIDE;
5 684 --
5 685 --      PROCEDURE TDialogTEView.ScrollSelectionIntoView; OVERRIDE;
5 686 --
5 687 --      ($IFC qDebug)
5 688 --      PROCEDURE TDialogTEView.Fields (PROCEDURE DoToField (fieldName: Str255;
5 689 --                                   fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
5 690 --      ($ENDC)
5 691 --      END (TDialogTEView);
5 692 --

```

```
5 693 -- {-----+
5 694 -- | Global routines |
5 695 -- +-----+
5 696 -- PROCEDURE InitUDialog;
5 697 --
5 698 -- IMPLEMENTATION
5 699 --
5 700 -- {$I UDialog.incl.p}
```

```

6 1 --      {SP}
6 2 --      { UDDialog.p }
6 3 --      { Copyright © 1988 Apple Computer Inc. All rights reserved. }
6 4 --
6 5 --      (*****
6 6 --      (*      Global Routines      *)
6 7 --      (*****
6 8 --      {$D DlgInit}
6 9 --      {-----+
6 10 --      |      InitUDialog      |
6 11 --      +-----}
6 12 -- A  PROCEDURE InitUDialog;
6 13 --      VAR
6 14 --          aDialogView:  TDialogView;
6 15 --          aControl:      TControl;
6 16 --          aCtlMgr:       TCtlMgr;
6 17 --          aButton:       TButton;
6 18 --          aCheckBox:     TCheckBox;
6 19 --          aRadio:        TRadio;
6 20 --          aCluster:      TCluster;
6 21 --          aIcon:         TIcon;
6 22 --          aPicture:      TPicture;
6 23 --          aPopup:        TPopup;
6 24 --          aStaticText:   TStaticText;
6 25 --          aEditText:      TEditText;
6 26 --          aNumberText:   TNumberText;
6 27 --          aPattern:      TPattern;
6 28 --
6 29 0- A  BEGIN
6 30 --      {$IFC qTemplateViews}
6 31 --          NEW(aDialogView);  FailNIL(aDialogView);  RegisterStdType('TDialogView', 'dlog', aDialogView);
6 32 --          NEW(aControl);      FailNIL(aControl);      RegisterStdType('TControl', 'cntl', aControl);
6 33 --          NEW(aCtlMgr);        FailNIL(aCtlMgr);        RegisterType('TCtlMgr', aCtlMgr);
6 34 --          NEW(aButton);        FailNIL(aButton);        RegisterStdType('TButton', 'btn', aButton);
6 35 --          NEW(aCheckBox);       FailNIL(aCheckBox);       RegisterStdType('TCheckBox', 'chkb', aCheckBox);
6 36 --          NEW(aRadio);         FailNIL(aRadio);         RegisterStdType('TRadio', 'radb', aRadio);
6 37 --          NEW(aCluster);       FailNIL(aCluster);       RegisterStdType('TCluster', 'clus', aCluster);
6 38 --          NEW(aIcon);          FailNIL(aIcon);          RegisterStdType('TIcon', 'icon', aIcon);
6 39 --          NEW(aPicture);       FailNIL(aPicture);       RegisterStdType('TPicture', 'pict', aPicture);
6 40 --          NEW(aPopup);         FailNIL(aPopup);         RegisterStdType('TPopup', 'popp', aPopup);
6 41 --          NEW(aStaticText);    FailNIL(aStaticText);    RegisterStdType('TStaticText', 'stat', aStaticText);
6 42 --          NEW(aEditText);      FailNIL(aEditText);      RegisterStdType('TEditText', 'edit', aEditText);
6 43 --          NEW(aNumberText);    FailNIL(aNumberText);    RegisterStdType('TNumberText', 'nmbr', aNumberText);
6 44 --          NEW(aPattern);       FailNIL(aPattern);       RegisterStdType('TPattern', 'patn', aPattern);
6 45 --      {$ENDC}
6 46 0- A  END;
6 47 --
6 48 --
6 49 --      {$S DlgRes}
6 50 --      {-----+
6 51 --      |      FlashControl      |
6 52 --      +-----}
6 53 -- A  PROCEDURE FlashControl (theControl: TControl);
6 54 --      VAR
6 55 --          dontCare:        LONGINT;
6 56 0- A  BEGIN
6 57 --      IF theControl <> NIL THEN
6 58 1-      BEGIN
6 59 --          theControl.HiliteState(TRUE, kRedraw);
6 60 --          Delay(8, dontCare);
6 61 --          theControl.HiliteState(FALSE, kRedraw);
6 62 1-      END;
6 63 0- A  END;
6 64 --
6 65 --      {$S DlgRes}
6 66 --      {-----+
6 67 --      |      GetIfBkColor      |
6 68 --      +-----}
6 69 -- A  PROCEDURE GetIfBkColor (VAR aColor: RGBColor);
6 70 --      CONST
6 71 --          BlackBit   = 5;
6 72 --          YellowBit  = 6;
6 73 --          MagentaBit  = 7;
6 74 --          CyanBit     = 8;
6 75 --      VAR
6 76 --          oldColor:   LONGINT;
6 77 0- A  BEGIN
6 78 --      IF qConfiguration.hasColorQD THEN
6 79 --          GetBackColor(aColor)
6 80 --      ELSE
6 81 1-      BEGIN
6 82 --          (*      xxxxxxxx C.MY B rgb w b - RGB      *)
6 83 --          blackColor   = 33 - 0000000 0.00 1 000 0 1 - 000
6 84 --          whiteColor   = 30 - 0000000 0.00 0 111 1 0 - 111
6 85 --          redColor     = 205 - 0000000 0.11 0 011 0 1 - 100
6 86 --          greenColor   = 341 - 0000000 1.01 0 101 0 1 - 010
6 87 --          blueColor    = 409 - 0000000 1.10 0 110 0 1 - 001
6 88 --          cyanColor    = 273 - 0000000 1.00 0 100 0 1 - 011
6 89 --          magentaColor = 137 - 0000000 0.10 0 010 0 1 - 101
6 90 --          yellowColor  = 69 - 0000000 0.01 0 001 0 1 - 110
6 91 --          *)
6 92 --          oldColor := thePort^.bkColor;  { Fetch old color }
6 93 --          aColor := gRGBBlack;           { Prime returned color to black }
6 94 --          IF BTST(oldColor, BlackBit) THEN { If color isn't black, force CMY = 111 }
6 95 --              oldColor := BOR(oldColor, $1C0);
6 96 --          IF NOT BTST(oldColor, CyanBit) THEN { Absence of cyan = presence of red }
6 97 --              aColor.red := $FFFF;
6 98 --          IF NOT BTST(oldColor, MagentaBit) THEN { Absence of magenta = presence of green }

```

```

6 99 --          aColor.green := $FFFF;
6 100 --          IF NOT BTST(oldColor, YellowBit) THEN { Absence of yellow = presence of blue }
6 101 --          aColor.blue := $FFFF;
6 102 -1          END;
6 103 -0 A      END;
6 104 --
6 105 --          {$S DlgRes}
6 106 --          {-----+
6 107 --          | SetIfBkColor |
6 108 --          +-----}
6 109 -- A      PROCEDURE SetIfBkColor (aColor: RGBColor);
6 110 --      CONST
6 111 --          SignBit = 15;
6 112 --      VAR
6 113 --          index: INTEGER;
6 114 --          oldColor: LONGINT;
6 115 0- A      BEGIN
6 116 --          IF gConfiguration.hasColorQD THEN
6 117 --              RGBBackColor(aColor)
6 118 --          ELSE
6 119 1-          BEGIN
6 120 --              index := 0;
6 121 --              IF BTST(aColor.red, SignBit) THEN { Prime index }
6 122 --                  index := 4; { Set bit if red >= $8000 }
6 123 --              IF BTST(aColor.green, SignBit) THEN { Set bit if green >= $8000 }
6 124 --                  index := index + 2;
6 125 --              IF BTST(aColor.blue, SignBit) THEN { Set bit if blue >= $8000 }
6 126 --                  index := index + 1;
6 127 2-          CASE index OF
6 128 --              0: oldColor := blackColor;
6 129 --              1: oldColor := blueColor;
6 130 --              2: oldColor := greenColor;
6 131 --              3: oldColor := cyanColor;
6 132 --              4: oldColor := redColor;
6 133 --              5: oldColor := magentaColor;
6 134 --              6: oldColor := yellowColor;
6 135 --              7: oldColor := whiteColor;
6 136 -2          END;
6 137 --          BackColor(oldColor);
6 138 -1          END;
6 139 -0 A      END;
6 140 --
6 141 --          {$S DlgRes}
6 142 --          {-----+ Determines fore and background colors, based on
6 143 --          | GetMenuColors | screen depth and which device it will come up on
6 144 --          +-----}
6 145 -- A      PROCEDURE GetMenuColors (popupRect: Rect; menuItem: INTEGER;
6 146 --                                VAR fColor, bColor: RGBColor);
6 147 --      VAR
6 148 --          gotTitle: BOOLEAN;
6 149 --          gdh: GDHandle;
6 150 --          mce: MCEntPtr;
6 151 --          titleMce: MCEnt;
6 152 --          globalMce: MCEnt;
6 153 --
6 154 --          {-----+
6 155 --          | SetBadColors |
6 156 --          +-----}
6 157 -- B      PROCEDURE SetBadColors;
6 158 --
6 159 0- B      BEGIN
6 160 --          fColor := gRGBBlack;
6 161 --          bColor := gRGBWhite;
6 162 -0 B      END;
6 163 --
6 164 0- A      BEGIN
6 165 --          gotTitle := FALSE; { Assume the worst. We always do. }
6 166 --
6 167 --          IF gConfiguration.hasColorQD THEN { First, be sure we have color QD... }
6 168 1-          BEGIN
6 169 --              LocalToGlobal(popupRect.topLeft); { Globalize rect, in focused coordinates }
6 170 --              LocalToGlobal(popupRect.botRight);
6 171 --              gdh := GetMaxDevice(popupRect); { Get device characteristics for that rect }
6 172 --
6 173 --              IF gdh^.gdPMap^.pixelSize > 1 THEN { If we have more than two colors }
6 174 2-              BEGIN
6 175 --                  mce := GetMCEnt(menuID, 0); { Always get title entry }
6 176 --                  IF mce <> NIL THEN
6 177 3-                  BEGIN
6 178 --                      gotTitle := TRUE;
6 179 --                      titleMce := mce; { Future calls could shift memory }
6 180 -3                  END;
6 181 --
6 182 --                  IF NOT gotTitle THEN { If we can't get the title entry, then... }
6 183 3-                  BEGIN
6 184 --                      mce := GetMCEnt(0, 0); { _we'll need the global entry, too }
6 185 --                      IF mce <> NIL THEN
6 186 --                          globalMce := mce^
6 187 --                      ELSE
6 188 4-                      BEGIN
6 189 --                          SetBadColors; { If no title, AND no global entry, punt }
6 190 --                          EXIT(GetMenuColors); { Even if item guy exists. No title, No washee }
6 191 -4                      END;
6 192 -3                      END;
6 193 --
6 194 --          { Handle a title color request }
6 195 --          IF itemNum = 0 THEN
6 196 3-          BEGIN
6 197 --              IF gotTitle THEN

```

```

6 198 4-          BEGIN
6 199 --          fColor := titleMce.mctRGB1;
6 200 --          bColor := titleMce.mctRGB2;
6 201 4-          END
6 202 --          ELSE { IF gotGlobal <<< has to be, by this point }
6 203 4-          BEGIN
6 204 --          fColor := globalMce.mctRGB1;
6 205 --          bColor := globalMce.mctRGB4;
6 206 4-          END;
6 207 -3          END
6 208 --          { Otherwise, it's for an item }
6 209 --          ELSE
6 210 3-          BEGIN
6 211 --          mce := GetMCEntree(menuID, itemNum);
6 212 --          IF mce <> NIL THEN
6 213 --              fColor := mce*.mctRGB2
6 214 --          ELSE IF gotTitle THEN
6 215 --              fColor := titleMce.mctRGB3
6 216 --          ELSE
6 217 --              fColor := globalMce.mctRGB3;
6 218 --
6 219 --          IF gotTitle THEN
6 220 --              bColor := titleMce.mctRGB4
6 221 --          ELSE
6 222 --              bColor := globalMce.mctRGB2;
6 223 -3          END;
6 224 -2          END
6 225 --          ELSE
6 226 --              SetBadColors;                { Only one bit depth. Default to B&W }
6 227 -1          END
6 228 --          ELSE
6 229 --              SetBadColors;                { Not using Color QuickDraw. B&W for sure }
6 230 --          { $IFC qDebug }
6 231 --          IF qIntenseDebugging THEN
6 232 1-          BEGIN
6 233 --              IF itemNum = 0 THEN WRITE('Title ') ELSE WRITE('Item #', itemNum:0);
6 234 --              Writeln(' foreground color- R:', fColor.red:0, ' G:', fColor.green:0,
6 235 --              ' B:', fColor.blue:0);
6 236 --              IF itemNum = 0 THEN WRITE('Title ') ELSE WRITE('Item #', itemNum:0);
6 237 --              Writeln(' background color- R:', bColor.red:0, ' G:', bColor.green:0,
6 238 --              ' B:', bColor.blue:0);
6 239 -1          END;
6 240 --          { $ENDC }
6 241 -0 A      END;
6 242 --
6 243 --
6 244 --          (*****
6 245 --          (*      T D i a l o g V i e w      *)
6 246 --          (*****
6 247 --
6 248 --          { $IFC qProceduralViews }
6 249 --          { $S DlgOpen }
6 250 --          {-----+
6 251 --          | IDialogView |
6 252 --          +-----+
6 253 -- A      PROCEDURE TDialogView.IDialogView (itsDocument: TDocument; itsSuperView: TView;
6 254 --          itsLocation, itsSize: VPoint;
6 255 --          itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 256 --          itsDefItemID, itsCancelItemID: IDType);
6 257 --          VAR
6 258 --              anAssociation: TAssociation;
6 259 --              aDialogTEView: TDialogTEView;
6 260 --              fi: FailInfo;
6 261 --
6 262 --          {-----+
6 263 --          | HandleFailure |
6 264 --          +-----+
6 265 -- B      PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 266 0- B      BEGIN
6 267 --          Free;
6 268 -0 B      END;
6 269 --
6 270 0- A      BEGIN
6 271 --          fParamTxt := NIL;                { In case of a catastrophe }
6 272 --          fTEView := NIL;                  { Ditto. }
6 273 --          IView(itsDocument, itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 274 --
6 275 --          CatchFailures(fi, HandleFailure);
6 276 --          NEW(anAssociation);                { Okay to allocate list now }
6 277 --          FailNIL(anAssociation);
6 278 --          anAssociation.IAssociation;
6 279 --          fParamTxt := anAssociation;
6 280 --          fDefaultItem := itsDefItemID;
6 281 --          fCancelItem := itsCancelItemID;
6 282 --          fCurrentEditText := NIL;
6 283 --          fDismissed := FALSE;
6 284 --          fDismitter := kNoIdentifier;
6 285 --
6 286 --          aDialogTEView := MakeTEView;
6 287 --          fTEView := aDialogTEView;
6 288 --          Success(fi);
6 289 -0 A      END;
6 290 --          { $ENDC }
6 291 --
6 292 --
6 293 --          { $IFC qTemplateViews }
6 294 --          { $S DlgOpen }
6 295 --          {-----+
6 296 --          | IRes |

```



```

6 297 -- +-----}
6 298 -- A PROCEDURE TDialogView.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
6 299 --
6 300 -- VAR
6 301 --     anAssociation:    TAssociation;
6 302 --     aDialogTEView:    TDialogTEView;
6 303 --     fi:                FailInfo;
6 304 --
6 305 --     {-----+
6 306 --     |   HandleFailure
6 307 --     +-----}
6 308 -- B PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 309 0- B BEGIN
6 310 --     Free;
6 311 -0 B END;
6 312 --
6 313 0- A BEGIN
6 314 --     fParamTxt := NIL;                { In case of a catastrophe }
6 315 --     fTEView := NIL;                { Ditto. }
6 316 --     INHERITED IRes(itsDocument, itsSuperview, itsParams);
6 317 --     WITH DialogViewTemplatePtr(itsParams)^ DO
6 318 1- BEGIN
6 319 --         fDefaultItem := defaultItem;
6 320 --         fCancelItem := cancelItem;
6 321 -1 END;
6 322 --     CatchFailures(fi, HandleFailure);
6 323 --     NEW(anAssociation);                { Okay to allocate list now }
6 324 --     FailNIL(anAssociation);
6 325 --     anAssociation.IAssociation;
6 326 --     fParamTxt := anAssociation;
6 327 --     fCurrentEditText := NIL;
6 328 --     fDismissed := FALSE;
6 329 --     fDismitter := kNoIdentifier;
6 330 --
6 331 --     aDialogTEView := MakeTEView;
6 332 --     fTEView := aDialogTEView;
6 333 --     Success(fi);
6 334 --
6 335 --     OffsetPtr(itsParams, SIZEOF(DialogViewTemplate));
6 336 -0 A END;
6 337 --     {$ENDC}
6 338 --
6 339 --
6 340 --     {$IFC qWriteTemplates}
6 341 --     {$S MAWriteRes}
6 342 --     {-----+
6 343 --     |   WRes
6 344 --     +-----}
6 345 -- A PROCEDURE TDialogView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 346 --
6 347 -- VAR
6 348 --     dgPtr:    DialogViewTemplatePtr;
6 349 --
6 350 0- A BEGIN
6 351 --     INHERITED WRes(theResource, itsParams);
6 352 --
6 353 --     dgPtr := DialogViewTemplatePtr(ExpandPtr(theResource, itsParams,
6 354 --         SIZEOF(DialogViewTemplate)));
6 355 --
6 356 --     WITH dgPtr^ DO
6 357 1- BEGIN
6 358 --         defaultItem := fDefaultItem;
6 359 --         cancelItem := fCancelItem;
6 360 -1 END;
6 361 -0 A END;
6 362 --
6 363 --     {$S MAWriteRes}
6 364 --     {-----+
6 365 --     |   WriteRes
6 366 --     +-----}
6 367 -- A PROCEDURE TDialogView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 368 0- A BEGIN
6 369 --     qWResSignature := 'dlog';    qWResType := 'TDialogView';
6 370 --     WRes(theResource, itsParams);
6 371 -0 A END;
6 372 --     {$ENDC}
6 373 --
6 374 --
6 375 --     {$S DlgClose}
6 376 --     {-----+
6 377 --     |   Free
6 378 --     +-----}
6 379 -- A PROCEDURE TDialogView.Free; OVERRIDE;
6 380 --
6 381 0- A BEGIN
6 382 --     FreeObject(fParamTxt);
6 383 --     FreeObject(fTEView);
6 384 --     INHERITED Free;
6 385 -0 A END;
6 386 --
6 387 --
6 388 --     {$S DlgRes}
6 389 --     {-----+
6 390 --     |   CanDismiss
6 391 --     +-----}
6 392 -- A FUNCTION TDialogView.CanDismiss (dismissing: IDType): BOOLEAN;
6 393 0- A BEGIN
6 394 --     CanDismiss := TRUE;
6 395 --     IF (LONGINT(fCancelItem) = LONGINT(kNoIdentifier)) | (dismissing <> fCancelItem) THEN

```

```

6 396 --      IF (fCurrentEditText <> NIL) & NOT fCurrentEditText.Validate THEN
6 397 1-      BEGIN
6 398 --      CanDismiss := FALSE;
6 399 --      CantDeselect(fCurrentEditText);
6 400 -1      END;
6 401 -0 A  END;
6 402 --
6 403 --
6 404 --      {$S DlgNonRes}
6 405 --      {-----+
6 406 --      |   CantDeselect   |
6 407 --      +-----}
6 408 -- A  PROCEDURE TDialogView.CantDeselect (theEditText: TEditText);
6 409 --
6 410 0- A  BEGIN
6 411 --      SysBeep(2);
6 412 --      theEditText.RestartEdit;
6 413 -0 A  END;
6 414 --
6 415 --
6 416 --      {$S DlgClose}
6 417 --      {-----+
6 418 --      |   Close         |
6 419 --      +-----}
6 420 -- A  PROCEDURE TDialogView.Close; OVERRIDE;
6 421 --
6 422 0- A  BEGIN
6 423 --      IF LONGINT(fDismitter) = LONGINT(kNoIdentifier) THEN
6 424 --      IF CanDismiss(fDefaultItem) THEN
6 425 --      DismissDialog(fDefaultItem)
6 426 --      ELSE
6 427 --      Failure(0, 0);          { Don't close the window }
6 428 -0 A  END;
6 429 --
6 430 --
6 431 --      {$S DlgClose}
6 432 --      {-----+
6 433 --      |   DismissDialog  |
6 434 --      +-----}
6 435 -- A  PROCEDURE TDialogView.DismissDialog (dismitter: IDType);
6 436 --
6 437 0- A  BEGIN
6 438 --      { Override to do whatever your dialog is supposed to end up doing }
6 439 --      fDismissed := TRUE;
6 440 --      fDismitter := dismitter;
6 441 -0 A  END;
6 442 --
6 443 --      {$S DlgRes}
6 444 --      {-----+
6 445 --      |   DoChoice       |
6 446 --      +-----}
6 447 -- A  PROCEDURE TDialogView.DoChoice (origView: TView; itsChoice: INTEGER); OVERRIDE;
6 448 0- A  BEGIN
6 449 1-      CASE itsChoice OF
6 450 --      mEditTextHit:
6 451 2-      BEGIN
6 452 --      ($IFC qDebug)
6 453 --      IF NOT Member(origView, TEditText) THEN
6 454 --      ProgramBreak(' Got mEditTextHit on non-TEditText view. ')
6 455 --      ELSE
6 456 --      {$ENDC}
6 457 --      DoSelectEditText(TEditText(origView), FALSE);
6 458 -2      END;
6 459 --      OTHERWISE
6 460 --      IF Member(origView, TControl) & TControl(origView).fDismissesDialog THEN
6 461 2-      BEGIN
6 462 --      IF CanDismiss(origView.fIdentifier) THEN
6 463 --      DismissDialog(origView.fIdentifier);
6 464 -2      END
6 465 --      ELSE
6 466 --      INHERITED DoChoice(origView, itsChoice);
6 467 -1      END;
6 468 -0 A  END;
6 469 --
6 470 --      {$S DlgRes}
6 471 --      {-----+
6 472 --      |   DoCommandKey   |
6 473 --      +-----}
6 474 -- A  FUNCTION TDialogView.DoCommandKey (ch: CHAR; VAR info: EventInfo): TCommand; OVERRIDE;
6 475 0- A  BEGIN
6 476 --      IF (ch = '.') AND (LONGINT(fCancelItem) <> LONGINT(kNoIdentifier)) THEN
6 477 1-      BEGIN
6 478 --      IF CanDismiss(fCancelItem) THEN
6 479 2-      BEGIN
6 480 --      FlashControl(TControl(FindSubView(fCancelItem)));
6 481 --      DismissDialog(fCancelItem);
6 482 -2      END;
6 483 --      DoCommandKey := gNoChanges;
6 484 -1      END
6 485 --      ELSE
6 486 --      DoCommandKey := INHERITED DoCommandKey(ch, info);
6 487 -0 A  END;
6 488 --
6 489 --      {$S DlgRes}
6 490 --      {-----+
6 491 --      |   DoKeyCommand   |
6 492 --      +-----}
6 493 -- A  FUNCTION TDialogView.DoKeyCommand (ch: Char; aKeyCode: INTEGER;
6 494 --      VAR info: EventInfo): TCommand; OVERRIDE;

```

```

6 495 0- A BEGIN
6 496 -- { If we get this far, nobody's handled the Tab, Enter, or Return keys, so we will }
6 497 -- IF ch = chTab THEN
6 498 --     DoKeyCommand := HandleTab(SELF, info)
6 499 -- ELSE IF (ch = chEnter) | (ch = chReturn) THEN
6 500 --     DoKeyCommand := HandleCR(SELF, info)
6 501 -- ELSE
6 502 --     DoKeyCommand := INHERITED DoKeyCommand(ch, aKeyCode, info);
6 503 -0 A END;
6 504 --
6 505 -- {$S DlgRes}
6 506 -- {-----+
6 507 -- | DoSelectEditText |
6 508 -- +-----}
6 509 - A PROCEDURE TDialogView.DoSelectEditText (theEditText: TEditText; selectChars: BOOLEAN);
6 510 --
6 511 -- VAR
6 512 --     itsWindow: TWindow;
6 513 --
6 514 0- A BEGIN
6 515 -- IF theEditText <> fCurrentEditText THEN { If we're not editing this view.. }
6 516 -- IF (fCurrentEditText <> NIL) & NOT fCurrentEditText.StopEdit THEN
6 517 --     CantDeselect(fCurrentEditText)
6 518 -- ELSE
6 519 1- BEGIN
6 520 --     fCurrentEditText := theEditText;
6 521 --     IF theEditText <> NIL THEN
6 522 --         theEditText.StartEdit(selectChars, fTEView)
6 523 --     ELSE
6 524 2- BEGIN
6 525 --         itsWindow := GetWindow; { Set the window's target, which sets }
6 526 --         IF itsWindow <> NIL THEN
6 527 --             itsWindow.SetTarget(SELF);
6 528 -2 END;
6 529 -1 END
6 530 -- ELSE IF selectChars THEN { Make sure all the chars are selected. }
6 531 --     theEditText.SetSelection(0, MAXINT, kRedraw);
6 532 -0 A END;
6 533 --
6 534 -- {$S DlgRes}
6 535 -- {-----+
6 536 -- | EachEditText |
6 537 -- +-----}
6 538 - A PROCEDURE TDialogView.EachEditText (PROCEDURE DoToEditText (theEditText: TEditText));
6 539 --
6 540 -- {-----+
6 541 -- | CheckSubView |
6 542 -- +-----}
6 543 - B PROCEDURE CheckSubView (theSubView: TView);
6 544 --
6 545 0- B BEGIN
6 546 -- IF MEMBER(theSubView, TEditText) THEN
6 547 --     DoToEditText(TEditText(theSubView))
6 548 -- ELSE
6 549 --     theSubView.EachSubView(CheckSubView);
6 550 -0 B END;
6 551 --
6 552 0- A BEGIN
6 553 --     EachSubView(CheckSubView);
6 554 -0 A END;
6 555 --
6 556 --
6 557 -- {$S DlgRes}
6 558 -- {-----+
6 559 -- | GetDialogView |
6 560 -- +-----}
6 561 - A FUNCTION TDialogView.GetDialogView: TView; OVERRIDE;
6 562 0- A BEGIN
6 563 --     GetDialogView := SELF;
6 564 -0 A END;
6 565 --
6 566 -- {$S DlgRes}
6 567 -- {-----+
6 568 -- | HandleCR |
6 569 -- +-----}
6 570 - A FUNCTION TDialogView.HandleCR (callingView: TView; VAR info: EventInfo): TCommand;
6 571 --
6 572 -- VAR
6 573 --     fDefaultControl: TControl;
6 574 --     dontCare: LONGINT;
6 575 0- A BEGIN
6 576 -- IF LONGINT(fDefaultItem) <> LONGINT(kNoIdentifier) THEN
6 577 1- BEGIN
6 578 --     FlashControl(TControl(FindSubView(fDefaultItem)));
6 579 --     DismissDialog(fDefaultItem);
6 580 -1 END;
6 581 -- HandleCR := gNoChanges;
6 582 -0 A END;
6 583 --
6 584 -- {$S DlgRes}
6 585 -- {-----+
6 586 -- | HandleTab |
6 587 -- +-----}
6 588 - A FUNCTION TDialogView.HandleTab (callingView: TView; VAR info: EventInfo): TCommand;
6 589 --
6 590 -- VAR
6 591 --     first: TEditText;
6 592 --     last: TEditText;
6 593 --     next: TEditText;

```

```

6 594 --      previous:      TEditText;
6 595 --
6 596 0- A BEGIN
6 597 --      SurveyEditText(first, last, next, previous);
6 598 --
6 599 --      IF info.theShiftKey THEN
6 600 --          next := previous;
6 601 --
6 602 --      IF next <> NIL THEN
6 603 --          DoSelectEditText(next, TRUE);
6 604 --      HandleTab := gNoChanges;
6 605 0- A END;
6 606 --
6 607 --      {$S DlgOpen}
6 608 --      {-----+
6 609 --      | MakeTEView |
6 610 --      +-----}
6 611 -- A FUNCTION TDialogView.MakeTEView: TDialogTEView;
6 612 --
6 613 --      VAR
6 614 --          aDialogTEView:      TDialogTEView;
6 615 --
6 616 0- A BEGIN
6 617 --      NEW(aDialogTEView);
6 618 --      FailNIL(aDialogTEView);
6 619 --      aDialogTEView.ITextView(NIL, SELF, gZeroVPt, gZeroVPt, sizeFixed, sizeFixed,
6 620 --          gZeroRect, gSystemStyle, teJustLeft, kWithoutStyle, FALSE);
6 621 --      aDialogTEView.fHTE^.clickLoop :=      { We want standard scrolling behavior }
6 622 --      defClickLoopProc;
6 623 --      aDialogTEView.fEditText := NIL;
6 624 --      aDialogTEView.AutoScrolling(TRUE);
6 625 --      MakeTEView := aDialogTEView;
6 626 0- A END;
6 627 --
6 628 --
6 629 --      {$S DlgRes}
6 630 --      {-----+
6 631 --      | ParamTxt |
6 632 --      +-----}
6 633 -- A PROCEDURE TDialogView.ParamTxt (keyStr, valueStr: Str255);
6 634 0- A BEGIN
6 635 --      fParamTxt.InsertEntry(keyStr, valueStr);
6 636 0- A END;
6 637 --
6 638 --
6 639 --      {$S DlgRes}
6 640 --      {-----+
6 641 --      | PoseModally |
6 642 --      +-----}
6 643 -- A FUNCTION TDialogView.PoseModally: IDType;
6 644 --
6 645 --      VAR
6 646 --          itsWindow:      TWindow;
6 647 --
6 648 0- A BEGIN
6 649 --      itsWindow := GetWindow;
6 650 --      IF itsWindow <> NIL THEN
6 651 1- BEGIN
6 652 --          itsWindow.Open;
6 653 --          fDismissed := FALSE;
6 654 2- REPEAT
6 655 --          gApplication.PollEvent;
6 656 2- UNTIL fDismissed;
6 657 --          PoseModally := fDismitter;
6 658 1- END
6 659 --      ELSE
6 660 --          PoseModally := kNoIdentifier;
6 661 0- A END;
6 662 --
6 663 --      {$S DlgRes}
6 664 --      {-----+
6 665 --      | ReplaceText |
6 666 --      +-----}
6 667 -- A PROCEDURE TDialogView.ReplaceText (theText: Handle);
6 668 --
6 669 --      {-----+
6 670 --      | ReplaceOnce |
6 671 --      +-----}
6 672 -- B PROCEDURE ReplaceOnce(item: TObjct);
6 673 --
6 674 --      VAR
6 675 --          offset:      LONGINT;
6 676 --          limit:      LONGINT;
6 676 0- B BEGIN
6 677 --          offset := 1;      { Prime starting value }
6 678 1- REPEAT
6 679 --          limit := GetHandleSize(theText) - 1;
6 680 --          WITH TEntry(item) DO
6 681 --              offset := Munger(theText, offset,
6 682 --                  Ptr(ORD4(fKey^)+1), LENGTH(fKey^), Ptr(ORD4(fValue^)+1), LENGTH(fValue^));
6 683 1- UNTIL (offset < 0) | (offset >= limit);
6 684 --          {$IFC gRangeCheck}{SR-}{SENDC}
6 685 --          theText^ := GetHandleSize(theText) - 1;
6 686 --          {$IFC gRangeCheck}{SR+}{SENDC}
6 687 0- B END;
6 688 --
6 689 0- A BEGIN
6 690 --      fParamTxt.fEntries.Each(ReplaceOnce);
6 691 0- A END;
6 692 --

```

```

6 693 --
6 694 --      {$S DlgRes}
6 695 --      {-----+
6 696 --      |   SelectEditText
6 697 --      +-----+}
6 698 -- A  PROCEDURE TDialogView.SelectEditText (itsIdentifier: IDType; selectChars: BOOLEAN);
6 699 --
6 700 --      VAR
6 701 --          aSubView:      TView;
6 702 --
6 703 -- A  BEGIN
6 704 --      aSubView := FindSubView(itsIdentifier);
6 705 --      IF (aSubView <> NIL) & (Member(aSubView, TEditText)) THEN
6 706 --          DoSelectEditText(TEditText(aSubView), selectChars);
6 707 -- A  END;
6 708 --
6 709 --      {$S DlgRes}
6 710 --      {-----+
6 711 --      |   SurveyEditText
6 712 --      +-----+}
6 713 -- A  PROCEDURE TDialogView.SurveyEditText (VAR first, last, next, previous: TEditText);
6 714 --
6 715 --      VAR
6 716 --          foundCurrent:      BOOLEAN;
6 717 --
6 718 --      {-----+
6 719 --      |   Survey
6 720 --      +-----+}
6 721 -- B  PROCEDURE Survey (theEditText: TEditText);
6 722 --
6 723 -- B  BEGIN
6 724 --      IF first = NIL THEN
6 725 --          first := theEditText;
6 726 --          last := theEditText;
6 727 --          IF theEditText = fCurrentEditText THEN
6 728 --              foundCurrent := TRUE
6 729 --          ELSE IF foundCurrent & (next = NIL) THEN
6 730 --              next := theEditText;
6 731 --              IF NOT foundCurrent THEN
6 732 --                  previous := theEditText;
6 733 -- B  END;
6 734 --
6 735 -- A  BEGIN
6 736 --      foundCurrent := FALSE;
6 737 --      next := NIL;
6 738 --      previous := NIL;
6 739 --      first := NIL;
6 740 --      last := NIL;
6 741 --      EachEditText(Survey);
6 742 --      IF next = NIL THEN
6 743 --          next := first;
6 744 --      IF previous = NIL THEN
6 745 --          previous := last;
6 746 -- A  END;
6 747 --
6 748 --
6 749 --      {$IFC qDebug}
6 750 --      {$S DlgFields}
6 751 --      {-----+
6 752 --      |   Fields
6 753 --      +-----+}
6 754 -- A  PROCEDURE TDialogView.Fields (PROCEDURE DoToField (fieldName: Str255;
6 755 --      fieldAddr: Ptr;
6 756 --      fieldType: INTEGER)); OVERRIDE;
6 757 -- A  BEGIN
6 758 --      DoToField('TDialogView', NIL, bClass);
6 759 --      DoToField('fDefaultItem', @fDefaultItem, bOSType);
6 760 --      DoToField('fCancelItem', @fCancelItem, bOSType);
6 761 --      DoToField('fParamTxt', @fParamTxt, bObject);
6 762 --      DoToField('fCurrentEditText', @fCurrentEditText, bObject);
6 763 --      DoToField('fTEView', @fTEView, bObject);
6 764 --      DoToField('fDismissed', @fDismissed, bBoolean);
6 765 --      DoToField('fDismitter', @fDismitter, bOSType);
6 766 --      INHERITED Fields(DoToField);
6 767 -- A  END;
6 768 --
6 769 --      {$ENDC}
6 770 --
6 771 --      (*****
6 772 --      (*       T B u t t o n
6 773 --      (*****
6 774 --
6 775 --      {$IFC qProceduralViews}
6 776 --      {$S DlgOpen}
6 777 --      {-----+
6 778 --      |   IButton
6 779 --      +-----+}
6 780 -- A  PROCEDURE TButton.IButton (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 781 --      itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 782 --      itsLabel: Str255);
6 783 -- A  BEGIN
6 784 --      ICtrlMgr(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet,
6 785 --      itsLabel, 0, 0, 0, pushButProc);
6 786 --      fDefChoice := mButtonHit;
6 787 -- A  END;
6 788 --      {$ENDC}
6 789 --
6 790 --
6 791 --      {$IFC qTemplateViews}

```

```

6 792 --      {$S DlgOpen}
6 793 --      {-----+
6 794 --      | IRes                                     |
6 795 --      +-----+}
6 796 -- A  PROCEDURE TButton.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 797 --
6 798 --      VAR
6 799 --          itsArea:      Rect;
6 800 --
6 801 -- A  BEGIN
6 802 --      INHERITED IRes(NIL, itsSuperView, itsParams);
6 803 --
6 804 --          fDefChoice := mButtonHit;
6 805 --          ControlArea(itsArea);
6 806 --          WITH ButtonTemplatePtr(itsParams)^ DO
6 807 --              CreateCMgrControl(itsArea, itsLabel, 0, 0, 0, pushButProc);
6 808 --
6 809 --              OffsetPtrWStr(itsParams, SIZEOF(ButtonTemplate));
6 810 -- A  END;
6 811 --      {$ENDC}
6 812 --
6 813 --
6 814 --      {$IFC qTemplateViews}
6 815 --      {$S MAWriteRes}
6 816 --      {-----+
6 817 --      | WRes                                     |
6 818 --      +-----+}
6 819 -- A  PROCEDURE TButton.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 820 --
6 821 --      VAR
6 822 --          theLabel:      Str255;
6 823 --          btPtr:         ButtonTemplatePtr;
6 824 --
6 825 -- A  BEGIN
6 826 --      INHERITED WRes(theResource, itsParams);
6 827 --
6 828 --          GetText(theLabel);
6 829 --
6 830 --          btPtr := ButtonTemplatePtr(ExpandPtrWStr(theResource, itsParams,
6 831 --              SIZEOF(ButtonTemplate), LENGTH(theLabel)));
6 832 --
6 833 --          (* btPtr^.itsLabel := theLabel; *)
6 834 --          CopyStr255(theLabel, PRStr(btPtr^.itsLabel));
6 835 -- A  END;
6 836 --      {$ENDC}
6 837 --
6 838 --
6 839 --      {$IFC qTemplateViews}
6 840 --      {$S MAWriteRes}
6 841 --      {-----+
6 842 --      | WriteRes                                     |
6 843 --      +-----+}
6 844 -- A  PROCEDURE TButton.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 845 -- A  BEGIN
6 846 --      gWResSignature := 'butn';    gWResType := 'TButton';
6 847 --      WRes(theResource, itsParams);
6 848 -- A  END;
6 849 --      {$ENDC}
6 850 --
6 851 --
6 852 --      (*****
6 853 --      (*      T C h e c k B o x
6 854 --      (*****
6 855 --
6 856 --      {$IFC qProceduralViews}
6 857 --      {$S DlgOpen}
6 858 --      {-----+
6 859 --      | ICheckBox                                     |
6 860 --      +-----+}
6 861 -- A  PROCEDURE TCheckBox.ICheckBox (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 862 --      itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 863 --      itsLabel: Str255; isTurnedOn: BOOLEAN);
6 864 -- A  BEGIN
6 865 --      ICtrlMgr(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet,
6 866 --          itsLabel, 0, 0, 1, checkBoxProc);
6 867 --      SetState(isTurnedOn, kDontRedraw);
6 868 --      fDefChoice := mCheckBoxHit;
6 869 -- A  END;
6 870 --      {$ENDC}
6 871 --
6 872 --
6 873 --      {$IFC qTemplateViews}
6 874 --      {$S DlgOpen}
6 875 --      {-----+
6 876 --      | IRes                                     |
6 877 --      +-----+}
6 878 -- A  PROCEDURE TCheckBox.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 879 --
6 880 --      VAR
6 881 --          itsArea:      Rect;
6 882 --
6 883 -- A  BEGIN
6 884 --      INHERITED IRes(NIL, itsSuperView, itsParams);
6 885 --
6 886 --          fDefChoice := mCheckBoxHit;
6 887 --          ControlArea(itsArea);
6 888 --          WITH CheckBoxTemplatePtr(itsParams)^ DO
6 889 --              CreateCMgrControl(itsArea, itsLabel, ORD(isOn), 0, 1, checkBoxProc);
6 890 --

```

```

6 891 --      OffsetPtrWStr(itsParams, SIZEOF(CheckBoxTemplate));
6 892 -0 A  END;
6 893 --      {$ENDC}
6 894 --
6 895 --
6 896 --      {$IFC qWriteTemplates}
6 897 --      {$SS MAWriteRes}
6 898 --      {-----+
6 899 --      | WRes
6 900 --      +-----}
6 901 - A  PROCEDURE TCheckBox.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 902 --
6 903 --      VAR
6 904 --          theLabel: Str255;
6 905 --          cbPtr: CheckBoxTemplatePtr;
6 906 --
6 907 0- A  BEGIN
6 908 --          INHERITED WRes(theResource, itsParams);
6 909 --
6 910 --          GetText(theLabel);
6 911 --
6 912 --          cbPtr := CheckBoxTemplatePtr(ExpandPtrWStr(theResource, itsParams,
6 913 --          SIZEOF(CheckBoxTemplate), LENGTH(theLabel)));
6 914 --
6 915 --          cbPtr^.isOn := IsOn;
6 916 --          (* cbPtr^.itsLabel := theLabel; *)
6 917 --          CopyStr255(theLabel, PRStr(cbPtr^.itsLabel));
6 918 -0 A  END;
6 919 --
6 920 --      {$SS MAWriteRes}
6 921 --      {-----+
6 922 --      | WriteRes
6 923 --      +-----}
6 924 - A  PROCEDURE TCheckBox.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 925 0- A  BEGIN
6 926 --          gWResSignature := 'chkb'; gWResType := 'TCheckBox';
6 927 --          WRes(theResource, itsParams);
6 928 -0 A  END;
6 929 --      {$ENDC}
6 930 --
6 931 --
6 932 --      {$SS DlgRes}
6 933 --      {-----+
6 934 --      | DoChoice
6 935 --      +-----}
6 936 - A  PROCEDURE TCheckBox.DoChoice(origView: TView; itsChoice: INTEGER);
6 937 0- A  BEGIN
6 938 --          IF itsChoice = mCheckBoxHit THEN
6 939 --              Toggle(kRedraw);
6 940 --          INHERITED DoChoice(origView, itsChoice);
6 941 -0 A  END;
6 942 --
6 943 --      {$SS DlgRes}
6 944 --      {-----+
6 945 --      | IsOn
6 946 --      +-----}
6 947 - A  FUNCTION TCheckBox.IsOn: BOOLEAN;
6 948 0- A  BEGIN
6 949 --          IsOn := GetVal <> 0;
6 950 -0 A  END;
6 951 --
6 952 --      {$SS DlgRes}
6 953 --      {-----+
6 954 --      | SetState
6 955 --      +-----}
6 956 - A  PROCEDURE TCheckBox.SetState (state, redraw: BOOLEAN);
6 957 0- A  BEGIN
6 958 --          SetVal(ORD(state), redraw);
6 959 -0 A  END;
6 960 --
6 961 --      {$SS DlgRes}
6 962 --      {-----+
6 963 --      | Toggle
6 964 --      +-----}
6 965 - A  PROCEDURE TCheckBox.Toggle (redraw: BOOLEAN);
6 966 0- A  BEGIN
6 967 --          SetVal(ORD(NOT IsOn), redraw);
6 968 -0 A  END;
6 969 --
6 970 --      {$SS DlgRes}
6 971 --      {-----+
6 972 --      | ToggleIf
6 973 --      +-----}
6 974 - A  PROCEDURE TCheckBox.ToggleIf (matchState, redraw: BOOLEAN);
6 975 0- A  BEGIN
6 976 --          IF IsOn = matchState THEN
6 977 --              SetVal(ORD(NOT IsOn), redraw);
6 978 -0 A  END;
6 979 --
6 980 --      (*****
6 981 --      (* T R a d i o
6 982 --      (*****
6 983 --
6 984 --      {$IFC qProceduralViews}
6 985 --      {$SS DlgOpen}
6 986 --      {-----+
6 987 --      | IRadio
6 988 --      +-----}
6 989 - A  PROCEDURE TRadio.IRadio (itsSuperview: TView; itsLocation, itsSize: VPoint;

```

```

6 990 --             itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 991 --             itsLabel: Str255; isTurnedOn: BOOLEAN);
6 992 0- A BEGIN
6 993 --         IctlMgr(itsSuperview, itsLocation, itsSize, itsHSizeDet, itsVSizeDet,
6 994 --             itsLabel, 0, 0, 1, radioButProc);
6 995 --         SetState(isTurnedOn, kDontRedraw);
6 996 --         fDefChoice := mRadioHit;
6 997 -0 A END;
6 998 --         {$ENDC}
6 999 --
6 1000 --
6 1001 --         {$IFC qTemplateViews}
6 1002 --         {$S DlgOpen}
6 1003 --         {-----+
6 1004 --         | IRes                                     |
6 1005 --         +-----+}
6 1006 -- A PROCEDURE TRadio.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
6 1007 --
6 1008 --         VAR
6 1009 --             itsArea: Rect;
6 1010 --
6 1011 0- A BEGIN
6 1012 --             INHERITED IRes(NIL, itsSuperview, itsParams);
6 1013 --
6 1014 --             fDefChoice := mRadioHit;
6 1015 --             ControlArea(itsArea);
6 1016 --             WITH RadioTemplatePtr(itsParams)^ DO
6 1017 --                 CreateCMgrControl(itsArea, itsLabel, ORD(isOn), 0, 1, radioButProc);
6 1018 --                 OffsetPtrWStr(itsParams, SIZEOF(RadioTemplate));
6 1019 -0 A END;
6 1020 --             {$ENDC}
6 1021 --
6 1022 --
6 1023 --         {$IFC qWriteTemplates}
6 1024 --         {$S MAWriteRes}
6 1025 --         {-----+
6 1026 --         | WRes                                     |
6 1027 --         +-----+}
6 1028 -- A PROCEDURE TRadio.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1029 --
6 1030 --         VAR
6 1031 --             theLabel: Str255;
6 1032 --             rdPtr: RadioTemplatePtr;
6 1033 --
6 1034 0- A BEGIN
6 1035 --             INHERITED WRes(theResource, itsParams);
6 1036 --
6 1037 --             GetText(theLabel);
6 1038 --
6 1039 --             rdPtr := RadioTemplatePtr(ExpandPtrWStr(theResource, itsParams,
6 1040 --                 SIZEOF(RadioTemplate), LENGTH(theLabel)));
6 1041 --
6 1042 --             rdPtr^.isOn := IsOn;
6 1043 --             (* rdPtr^.itsLabel := theLabel; *)
6 1044 --             CopyStr255(theLabel, PRStr(rdPtr^.itsLabel));
6 1045 -0 A END;
6 1046 --
6 1047 --         {$S MAWriteRes}
6 1048 --         {-----+
6 1049 --         | WriteRes                                   |
6 1050 --         +-----+}
6 1051 -- A PROCEDURE TRadio.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1052 0- A BEGIN
6 1053 --             qWResSignature := 'radb'; qWResType := 'TRadio';
6 1054 --             WRes(theResource, itsParams);
6 1055 -0 A END;
6 1056 --             {$ENDC}
6 1057 --
6 1058 --
6 1059 --         {$S DlgRes}
6 1060 --         {-----+
6 1061 --         | DoChoice                                   |
6 1062 --         +-----+}
6 1063 -- A PROCEDURE TRadio.DoChoice(origView: TView; itsChoice: INTEGER);
6 1064 0- A BEGIN
6 1065 --             IF (itsChoice = mRadioHit) & NOT IsOn THEN
6 1066 1- BEGIN
6 1067 --                 Toggle(kRedraw);
6 1068 --                 INHERITED DoChoice(origView, itsChoice); { Should this be called regardless??? }
6 1069 -1 END;
6 1070 -0 A END;
6 1071 --
6 1072 --         {$S DlgRes}
6 1073 --         {-----+
6 1074 --         | IsOn                                       |
6 1075 --         +-----+}
6 1076 -- A FUNCTION TRadio.IsOn: BOOLEAN;
6 1077 0- A BEGIN
6 1078 --             IsOn := GetVal <> 0;
6 1079 -0 A END;
6 1080 --
6 1081 --         {$S DlgRes}
6 1082 --         {-----+
6 1083 --         | SetState                                   |
6 1084 --         +-----+}
6 1085 -- A PROCEDURE TRadio.SetState (state, redraw: BOOLEAN);
6 1086 0- A BEGIN
6 1087 --             SetVal(ORD(state), redraw);
6 1088 -0 A END;

```



```

6 1089 --
6 1090 --      {$S DlgRes}
6 1091 --      {-----+
6 1092 --      | Toggle |
6 1093 --      +-----+
6 1094 -- A  PROCEDURE TRadio.Toggle (redraw: BOOLEAN);
6 1095 -- A  BEGIN
6 1096 --      SetVal(ORD(NOT IsOn), redraw);
6 1097 -- A  END;
6 1098 --
6 1099 --      {$S DlgRes}
6 1100 --      {-----+
6 1101 --      | ToggleIf |
6 1102 --      +-----+
6 1103 -- A  PROCEDURE TRadio.ToggleIf (matchState, redraw: BOOLEAN);
6 1104 -- A  BEGIN
6 1105 --      IF IsOn = matchState THEN
6 1106 --          SetVal(ORD(NOT IsOn), redraw);
6 1107 -- A  END;
6 1108 --
6 1109 --
6 1110 --      (*****
6 1111 --      (*      T C l u s t e r      *)
6 1112 --      (*****
6 1113 --
6 1114 --      {$IFC qProceduralViews}
6 1115 --      {$S DlgOpen}
6 1116 --      {-----+
6 1117 --      | ICluster |
6 1118 --      +-----+
6 1119 -- A  PROCEDURE TCluster.ICluster (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 1120 --                                itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 1121 --                                itsRsrcID, itsIndex: INTEGER);
6 1122 --      VAR
6 1123 --          aString:      Str255;
6 1124 --          fi:           FailInfo;
6 1125 --
6 1126 --      {-----+
6 1127 --      | HandleFailure |
6 1128 --      +-----+
6 1129 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1130 -- B  BEGIN
6 1131 --      Free;
6 1132 -- B  END;
6 1133 --
6 1134 -- A  BEGIN
6 1135 --      fDataHandle := NIL;
6 1136 --      IControl(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 1137 --      fRsrcID := itsRsrcID;
6 1138 --      fIndex := itsIndex;
6 1139 --      IF fRsrcID <> kNoResource THEN
6 1140 --          BEGIN
6 1141 --              CatchFailures(fi, HandleFailure);
6 1142 --              GetIndString(aString, fRsrcID, fIndex);
6 1143 --              FailResError;
6 1144 --              Success(fi);
6 1145 --              SetLabel(aString, kDontRedraw);
6 1146 --          END;
6 1147 --      ViewEnable(FALSE, kDontRedraw); { Default is not to enable hit testing }
6 1148 --      fDefChoice := mClusterHit;
6 1149 -- A  END;
6 1150 --      {$ENDC}
6 1151 --
6 1152 --
6 1153 --      {$IFC qTemplateViews}
6 1154 --      {$S DlgOpen}
6 1155 --      {-----+
6 1156 --      | IRes |
6 1157 --      +-----+
6 1158 -- A  PROCEDURE TCluster.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 1159 --
6 1160 -- A  BEGIN
6 1161 --      fDataHandle := NIL;
6 1162 --      INHERITED IRes(NIL, itsSuperView, itsParams);
6 1163 --      fDefChoice := mClusterHit;
6 1164 --
6 1165 --      WITH ClusterTemplatePtr(itsParams)^ DO
6 1166 --          SetLabel(itsLabel, kDontRedraw);
6 1167 --
6 1168 --      OffsetPtrWStr(itsParams, SIZEOF(ClusterTemplate));
6 1169 -- A  END;
6 1170 --      {$ENDC}
6 1171 --
6 1172 --
6 1173 --      {$IFC qWriteTemplates}
6 1174 --      {$S MAWriteRes}
6 1175 --      {-----+
6 1176 --      | WRes |
6 1177 --      +-----+
6 1178 -- A  PROCEDURE TCluster.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1179 --
6 1180 --      VAR
6 1181 --          theLabel:      Str255;
6 1182 --          clPtr:         ClusterTemplatePtr;
6 1183 --
6 1184 -- A  BEGIN
6 1185 --      INHERITED WRes(theResource, itsParams);
6 1186 --
6 1187 --      GetLabel(theLabel);

```

```

6 1188 --
6 1189 --      clPtr := ClusterTemplatePtr(ExpandPtrWStr(theResource, itsParams,
6 1190 --      SIZEOF(ClusterTemplate), LENGTH(theLabel)));
6 1191 --
6 1192 --      (* clPtr^.itsLabel := theLabel; *)
6 1193 --      CopyStr255(theLabel, PRStr(clPtr^.itsLabel));
6 1194 -0 A  END;
6 1195 --
6 1196 --      {$S MWriteRes}
6 1197 --      {-----+
6 1198 --      |   WriteRes
6 1199 --      +-----}
6 1200 -- A  PROCEDURE TCluster.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1201 0- A  BEGIN
6 1202 --      gWResSignature := 'clus';   gWResType := 'TCluster';
6 1203 --      WRes(theResource, itsParams);
6 1204 -0 A  END;
6 1205 --      {$ENDC}
6 1206 --
6 1207 --
6 1208 --      {$S DlgClose}
6 1209 --      {-----+
6 1210 --      |   Free
6 1211 --      +-----}
6 1212 -- A  PROCEDURE TCluster.Free; OVERRIDE;
6 1213 0- A  BEGIN
6 1214 --      ReleaseLabel;
6 1215 --      INHERITED Free;
6 1216 -0 A  END;
6 1217 --
6 1218 --      {$S DlgRes}
6 1219 --      {-----+
6 1220 --      |   DoChoice
6 1221 --      +-----}
6 1222 -- A  PROCEDURE TCluster.DoChoice (origView: View; itsChoice: INTEGER); OVERRIDE;
6 1223 --
6 1224 --      {-----+
6 1225 --      |   ResetRadios
6 1226 --      +-----}
6 1227 -- B  PROCEDURE ResetRadios(aView: TView);
6 1228 0- B  BEGIN
6 1229 --      IF MEMBER(aView, TRadio) AND      { If the subview is a TRadio, and... }
6 1230 --      (aView <> origView) THEN          { ...it's not the calling radio... }
6 1231 --      TRadio(aView).SetState(FALSE, kRedraw); { ...set it off and redraw it }
6 1232 -0 B  END;
6 1233 --
6 1234 0- A  BEGIN
6 1235 --      IF (itsChoice = mRadioHit) AND      { If we got this far, a radio's changed state }
6 1236 --      (origView.fSuperview = SELF) THEN  { Only worry about it if it's _our_ subview! }
6 1237 --      EachSubView(ResetRadios);          { Reset everybody except the calling radio }
6 1238 --      INHERITED DoChoice(origView, itsChoice);
6 1239 -0 A  END;
6 1240 --
6 1241 --      {$S DlgRes}
6 1242 --      {-----+
6 1243 --      |   Draw
6 1244 --      +-----}
6 1245 -- A  PROCEDURE TCluster.Draw (area: Rect); OVERRIDE;
6 1246 --      VAR
6 1247 --      fontHt:      INTEGER;
6 1248 --      labelWd:      INTEGER;
6 1249 --      oldTop:        INTEGER;
6 1250 --      err:           OSErr;
6 1251 --      fInfo:         FontInfo;
6 1252 --      theFrame:      Rect;
6 1253 --      realText:      StringHandle;
6 1254 --      aDialogView:   TDialogView;
6 1255 --      oldColor:      RGBColor;
6 1256 --
6 1257 0- A  BEGIN
6 1258 --      IF fDataHandle <> NIL THEN
6 1259 1-  BEGIN
6 1260 --      PenNormal; { Normalcy in most things }
6 1261 --      PenSize(fPenSize.h, fPenSize.v); (* PenSize(2, 2); *)
6 1262 --      GetIfColor(oldColor); { Save the original pen color }
6 1263 --      SetPortTextStyle(fTextStyle); { Install the object's TextStyle }
6 1264 --      GetFontInfo(fInfo); { Determine label's height }
6 1265 --      fontHt := fInfo.ascent + fInfo.descent + fInfo.leading;
6 1266 --      ControlArea(theFrame); { Get the control's extent }
6 1267 --      oldTop := theFrame.top;
6 1268 --      InsetRect(theFrame, { Inset the frame a little }
6 1269 --      fPenSize.h + 1, fPenSize.v + 1);
6 1270 --      theFrame.top := oldTop + BSR(fontHt, 1); { Bump top so it cuts label in half }
6 1271 --
6 1272 --      FrameRect(theFrame); { Draw the frame }
6 1273 --
6 1274 --      realText := fDataHandle; { Make a copy of the original text }
6 1275 --      err := HandToHand(Handle(realText)); { We won't die just because we can't copy }
6 1276 --      IF err <> noErr THEN { We'll just use original in an emergency }
6 1277 --      realText := fDataHandle
6 1278 --      ELSE
6 1279 2-  BEGIN
6 1280 --      aDialogView := TDialogView(GetDialogView);
6 1281 --      IF aDialogView <> NIL THEN
6 1282 --      aDialogView.ReplaceText(Handle(realText));
6 1283 -2  END;
6 1284 --      labelWd := StringWidth(realText^^) + 8;
6 1285 --      SetRect(theFrame, 16, 0, labelWd + 16, fontHt);
6 1286 --      HLock(Handle(realText));

```

```

6 1287 --      TextBox(Ptr(ORD4(realText^) + 1), { Draw the label
6 1288 --      LENGTH(realText^^), theFrame, teJustCenter);
6 1289 --      HUnlock(Handle(realText));
6 1290 --      IF err = noErr THEN { Dispose the handle if we could create it
6 1291 --          DisposHandle(Handle(realText));
6 1292 --      SetIfColor(oldColor);
6 1293 --      END;
6 1294 --      INHERITED Draw(area); { Let parents have a chance to draw too
6 1295 -- A  END;
6 1296 --
6 1297 --      {$S DlgNonRes}
6 1298 --      {-----+
6 1299 --      | GetLabel
6 1300 --      +-----+}
6 1301 -- A  PROCEDURE TCluster.GetLabel (VAR theLabel: Str255);
6 1302 -- A  BEGIN
6 1303 --      IF fDataHandle <> NIL THEN
6 1304 --          theLabel := fDataHandle^^
6 1305 --      ELSE
6 1306 --          theLabel := '';
6 1307 -- A  END;
6 1308 --
6 1309 --      {$S DlgNonRes}
6 1310 --      {-----+
6 1311 --      | ReleaseLabel
6 1312 --      +-----+}
6 1313 -- A  PROCEDURE TCluster.ReleaseLabel;
6 1314 -- A  BEGIN
6 1315 --      DisposIfHandle(fDataHandle);
6 1316 --      fRsrcID := kNoResource;
6 1317 --      fDataHandle := NIL;
6 1318 -- A  END;
6 1319 --
6 1320 --      {$S DlgRes}
6 1321 --      {-----+
6 1322 --      | ReportCurrent
6 1323 --      +-----+}
6 1324 -- A  FUNCTION TCluster.ReportCurrent: IDType;
6 1325 --      VAR
6 1326 --          rView: TView;
6 1327 --
6 1328 --      {-----+
6 1329 --      | FindRadio
6 1330 --      +-----+}
6 1331 -- B  FUNCTION FindRadio(aView: TView): BOOLEAN;
6 1332 -- B  BEGIN
6 1333 --      FindRadio := MEMBER(aView, TRadio) AND TRadio(aView).IsOn;
6 1334 -- B  END;
6 1335 --
6 1336 -- A  BEGIN
6 1337 --      rView := FirstSubViewThat(FindRadio);
6 1338 --      IF rView <> NIL THEN
6 1339 --          ReportCurrent := rView.fIdentifier
6 1340 --      ELSE
6 1341 --          ReportCurrent := kNoIdentifier;
6 1342 -- A  END;
6 1343 --
6 1344 --      {$S DlgNonRes}
6 1345 --      {-----+
6 1346 --      | SetLabel
6 1347 --      +-----+}
6 1348 -- A  PROCEDURE TCluster.SetLabel(theLabel: Str255; redraw: BOOLEAN);
6 1349 --
6 1350 -- A  BEGIN
6 1351 --      ReleaseLabel;
6 1352 --      IF theLabel <> '' THEN
6 1353 --          BEGIN
6 1354 --              fDataHandle := NewString(theLabel);
6 1355 --              IF MemError <> noErr THEN
6 1356 --                  fDataHandle := NIL;
6 1357 --          END;
6 1358 --      IF redraw THEN
6 1359 --          ForceRedraw;
6 1360 -- A  END;
6 1361 --
6 1362 --      {$IFC qDebug}
6 1363 --      {$S DlgFields}
6 1364 --      {-----+
6 1365 --      | Fields
6 1366 --      +-----+}
6 1367 -- A  PROCEDURE TCluster.Fields (PROCEDURE DoToField (fieldName: Str255;
6 1368 --      fieldAddr: Ptr;
6 1369 --      fieldType: INTEGER)); OVERRIDE;
6 1370 -- A  BEGIN
6 1371 --      DoToField('TCluster', NIL, bClass);
6 1372 --      DoToField('fRsrcID', @fRsrcID, bInteger);
6 1373 --      DoToField('fIndex', @fIndex, bInteger);
6 1374 --      DoToField('fDataHandle', @fDataHandle, bHandle);
6 1375 --      INHERITED Fields(DoToField);
6 1376 -- A  END;
6 1377 --
6 1378 --      {$ENDC}
6 1379 --
6 1380 --
6 1381 --      (*****
6 1382 --      (* T I C o n *)
6 1383 --      (*****
6 1384 --
6 1385 --      {$IFC qProceduralViews}

```

```

6 1386 --      {$S DlgOpen}
6 1387 --      {-----+
6 1388 --      |      IIcon
6 1389 --      +-----+}
6 1390 -- A  PROCEDURE TIcon.IIcon (itsSuperview: TView; itsLocation, itsSize: VPoint;
6 1391 --      itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 1392 --      itsRsrcID: INTEGER; preferColor: BOOLEAN);
6 1393 --      VAR
6 1394 --          fi:          FailInfo;
6 1395 --
6 1396 --      {-----+
6 1397 --      |      HandleFailure
6 1398 --      +-----+}
6 1399 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1400 -- B  BEGIN
6 1401 --      Free;
6 1402 -- B  END;
6 1403 --
6 1404 -- A  BEGIN
6 1405 --      fDataHandle := NIL;
6 1406 --      IControl(itsSuperview, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 1407 --      fPreferColor := preferColor;
6 1408 --      fRsrcID := itsRsrcID;
6 1409 --      IF fRsrcID <> kNoResource THEN
6 1410 --          BEGIN
6 1411 --              CatchFailures(fi, HandleFailure);
6 1412 --              IF fPreferColor AND gConfiguration.hasColorQD THEN
6 1413 --                  fDataHandle := Handle(GetCIcon(fRsrcID));
6 1414 --              IF fDataHandle = NIL THEN
6 1415 --                  BEGIN
6 1416 --                      fPreferColor := NOT kPreferColor;  ( Either can't or won't )
6 1417 --                      fDataHandle := GetIcon(fRsrcID);
6 1418 --                  END;
6 1419 --                  FailResError;
6 1420 --                  Success(fi);
6 1421 --              END;
6 1422 --              ViewEnable(FALSE, kDontRedraw);  ( Default is to not enable hit testing )
6 1423 --              fDefChoice := mIconHit;
6 1424 -- A  END;
6 1425 --      {$ENDC}
6 1426 --
6 1427 --
6 1428 --      {$IFC qTemplateViews}
6 1429 --      {$S DlgOpen}
6 1430 --      {-----+
6 1431 --      |      IRes
6 1432 --      +-----+}
6 1433 -- A  PROCEDURE TIcon.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
6 1434 --
6 1435 --      VAR
6 1436 --          fi:          FailInfo;
6 1437 --
6 1438 --      {-----+
6 1439 --      |      HandleFailure
6 1440 --      +-----+}
6 1441 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1442 -- B  BEGIN
6 1443 --      Free;
6 1444 -- B  END;
6 1445 --
6 1446 -- A  BEGIN
6 1447 --      fDataHandle := NIL;
6 1448 --      INHERITED IRes(NIL, itsSuperview, itsParams);
6 1449 --
6 1450 --      WITH IconTemplatePtr(itsParams)^ DO
6 1451 --          BEGIN
6 1452 --              fPreferColor := preferColor;
6 1453 --              fRsrcID := rsrcID;
6 1454 --          END;
6 1455 --          IF fRsrcID <> kNoResource THEN
6 1456 --              BEGIN
6 1457 --                  CatchFailures(fi, HandleFailure);
6 1458 --                  IF fPreferColor AND gConfiguration.hasColorQD THEN
6 1459 --                      fDataHandle := Handle(GetCIcon(fRsrcID));
6 1460 --                  IF fDataHandle = NIL THEN
6 1461 --                      BEGIN
6 1462 --                          fPreferColor := NOT kPreferColor;  ( Either can't or won't )
6 1463 --                          fDataHandle := GetIcon(fRsrcID);
6 1464 --                      END;
6 1465 --                      { Don't die because resource not found - just return NIL handle }
6 1466 --                      FailResError;
6 1467 --                      Success(fi);
6 1468 --                  END;
6 1469 --                  fDefChoice := mIconHit;
6 1470 --
6 1471 --                  OffsetPtr(itsParams, SIZEOF(IconTemplate));
6 1472 -- A  END;
6 1473 --      {$ENDC}
6 1474 --
6 1475 --
6 1476 --      {$IFC qWriteTemplates}
6 1477 --      {$S MAWriteRes}
6 1478 --      {-----+
6 1479 --      |      WRes
6 1480 --      +-----+}
6 1481 -- A  PROCEDURE TIcon.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1482 --
6 1483 --      VAR
6 1484 --          icPtr:      IconTemplatePtr;

```

```

6 1485 --
6 1486 0- A BEGIN
6 1487 --     INHERITED WRes(theResource, itsParams);
6 1488 --
6 1489 --     icPtr := IconTemplatePtr(ExpandPtr(theResource, itsParams, SIZEOF(IconTemplate)));
6 1490 --
6 1491 --     WITH icPtr^ DO
6 1492 1- BEGIN
6 1493 --         preferColor := fPreferColor;
6 1494 --     {$IFC qDebug}
6 1495 --         IF fRsrcID = kNoResource THEN
6 1496 --             WriteLn('Tried to write TIcon with no resource ID.');
```

```

6 1497 --     {$ENDC}
6 1498 --         rsrcID := fRsrcID;
6 1499 -1 END;
6 1500 -0 A END;
6 1501 --     {$ENDC}
6 1502 --
6 1503 --
6 1504 --     {$IFC qWriteTemplates}
6 1505 --     {$S MAWriteRes}
6 1506 --     {-----+
6 1507 --     | WriteRes |
6 1508 --     +-----}
6 1509 -- A PROCEDURE TIcon.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1510 0- A BEGIN
6 1511 --     gWResSignature := 'icon'; gWResType := 'TIcon';
6 1512 --     WRes(theResource, itsParams);
6 1513 -0 A END;
6 1514 --     {$ENDC}
6 1515 --
6 1516 --
6 1517 --     {$S DlgClose}
6 1518 --     {-----+
6 1519 --     | Free |
6 1520 --     +-----}
6 1521 -- A PROCEDURE TIcon.Free; OVERRIDE;
6 1522 0- A BEGIN
6 1523 --     ReleaseIcon;
6 1524 --     INHERITED Free;
6 1525 -0 A END;
6 1526 --
6 1527 --     {$S DlgRes}
6 1528 --     {-----+
6 1529 --     | Draw |
6 1530 --     +-----}
6 1531 -- A PROCEDURE TIcon.Draw(area: Rect); OVERRIDE;
6 1532 -- VAR
6 1533 --     oldState: SignedByte;
6 1534 --     theRect: Rect;
6 1535 --
6 1536 0- A BEGIN
6 1537 --     IF fDataHandle <> NIL THEN
6 1538 1- BEGIN
6 1539 --         PenNormal; { NECESSARY? }
6 1540 --         ControlArea(theRect);
6 1541 --         IF fRsrcID <> kNoResource THEN
6 1542 --             LoadResource(fDataHandle);
6 1543 --         IF fDataHandle = NIL THEN
6 1544 --             ReleaseIcon
6 1545 --         ELSE IF NOT fPreferColor THEN
6 1546 2- BEGIN
6 1547 --             oldState := GetHandleBits(fDataHandle);
6 1548 --             HNPurge(fDataHandle);
6 1549 --             PlotIcon(theRect, fDataHandle);
6 1550 --             SetHandleBits(fDataHandle, oldState);
6 1551 -2 END
6 1552 --         ELSE
6 1553 --             PlotCIcon(theRect, CIconHandle(fDataHandle));
6 1554 -1 END;
6 1555 --
6 1556 --     INHERITED Draw(area);
6 1557 -0 A END;
6 1558 --
6 1559 --     {$S DlgNonRes}
6 1560 --     {-----+
6 1561 --     | ReleaseIcon |
6 1562 --     +-----}
6 1563 -- A PROCEDURE TIcon.ReleaseIcon;
6 1564 0- A BEGIN
6 1565 --     fRsrcID := kNoResource;
6 1566 --     IF fPreferColor THEN
6 1567 --         DisposCIcon(CIconHandle(fDataHandle))
6 1568 --     ELSE
6 1569 --         HPurge(fDataHandle);
6 1570 --     fDataHandle := NIL;
6 1571 -0 A END;
6 1572 --
6 1573 --     {$S DlgNonRes}
6 1574 --     {-----+
6 1575 --     | SetIcon |
6 1576 --     +-----}
6 1577 -- A PROCEDURE TIcon.SetIcon(theIcon: Handle; redraw: BOOLEAN);
6 1578 0- A BEGIN
6 1579 --     ReleaseIcon;
6 1580 --     fDataHandle := theIcon;
6 1581 --     IF redraw THEN
6 1582 --         ForceRedraw;
6 1583 -0 A END;

```

```

6 1584 --
6 1585 --      ($IFC qDebug)
6 1586 --      ($S DlgFields)
6 1587 --      {-----+
6 1588 --      |   Fields   |
6 1589 --      +-----+
6 1590 -- A      PROCEDURE TIcon.Fields (PROCEDURE DoToField (fieldName: Str255;
6 1591 --                                fieldAddr: Ptr;
6 1592 --                                fieldType: INTEGER)); OVERRIDE;
6 1593 0- A      BEGIN
6 1594 --          DoToField('Ticon', NIL, bClass);
6 1595 --          DoToField('fPreferColor', @fPreferColor, bBoolean);
6 1596 --          DoToField('fRsrcID', @fRsrcID, bInteger);
6 1597 --          DoToField('fDataHandle', @fDataHandle, bHandle);
6 1598 --          INHERITED Fields(DoToField);
6 1599 0- A      END;
6 1600 --
6 1601 --      ($ENDC)
6 1602 --
6 1603 --
6 1604 --      (*****
6 1605 --      (*       T P a t t e r n
6 1606 --      (*****
6 1607 --
6 1608 --      ($IFC qProceduralViews)
6 1609 --      ($S DlgOpen)
6 1610 --      {-----+
6 1611 --      |   IPattern   |
6 1612 --      +-----+
6 1613 -- A      PROCEDURE TPattern.IPattern (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 1614 --                                itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 1615 --                                itsRsrcID: INTEGER; preferColor: BOOLEAN);
6 1616 --      VAR
6 1617 --          fi:          FailInfo;
6 1618 --
6 1619 --          {-----+
6 1620 --          |   HandleFailure   |
6 1621 --          +-----+
6 1622 -- B      PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1623 0- B      BEGIN
6 1624 --          Free;
6 1625 0- B      END;
6 1626 --
6 1627 0- A      BEGIN
6 1628 --          fDataHandle := NIL;
6 1629 --          IControl(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 1630 --          fPreferColor := preferColor;
6 1631 --          fRsrcID := itsRsrcID;
6 1632 --          IF fRsrcID <> kNoResource THEN
6 1633 1-      BEGIN
6 1634 --          CatchFailures(fi, HandleFailure);
6 1635 --          IF fPreferColor AND gConfiguration.hasColorQD THEN
6 1636 --              fDataHandle := Handle(GetPixPat(fRsrcID));
6 1637 --          IF fDataHandle = NIL THEN
6 1638 2-      BEGIN
6 1639 --              fPreferColor := NOT kPreferColor; { Either can't or won't }
6 1640 --              fDataHandle := Handle(GetPattern(fRsrcID));
6 1641 -2      END;
6 1642 --          FailResError;
6 1643 --          Success(fi);
6 1644 -1      END;
6 1645 --          ViewEnable(FALSE, kDontRedraw); { Default is to not enable hit testing }
6 1646 --          fDefChoice := mPatternHit;
6 1647 0- A      END;
6 1648 --      ($ENDC)
6 1649 --
6 1650 --
6 1651 --      ($IFC qTemplateViews)
6 1652 --      ($S DlgOpen)
6 1653 --      {-----+
6 1654 --      |   IRes   |
6 1655 --      +-----+
6 1656 -- A      PROCEDURE TPattern.IRes (itsDocument: TDocument; itsSuperView: TView;
6 1657 --                                VAR itsParams: Ptr); OVERRIDE;
6 1658 --      VAR
6 1659 --          fi:          FailInfo;
6 1660 --
6 1661 --          {-----+
6 1662 --          |   HandleFailure   |
6 1663 --          +-----+
6 1664 -- B      PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1665 0- B      BEGIN
6 1666 --          Free;
6 1667 0- B      END;
6 1668 --
6 1669 0- A      BEGIN
6 1670 --          fDataHandle := NIL;
6 1671 --          INHERITED IRes(NIL, itsSuperView, itsParams);
6 1672 --
6 1673 --          WITH PatternTemplatePtr(itsParams)^ DO
6 1674 1-      BEGIN
6 1675 --          fPreferColor := preferColor;
6 1676 --          fRsrcID := rsrcID;
6 1677 -1      END;
6 1678 --          IF fRsrcID <> kNoResource THEN
6 1679 1-      BEGIN
6 1680 --          CatchFailures(fi, HandleFailure);
6 1681 --          IF fPreferColor AND gConfiguration.hasColorQD THEN
6 1682 --              fDataHandle := Handle(GetPixPat(fRsrcID));

```

```

6 1683 --      IF fDataHandle = NIL THEN
6 1684 2-      BEGIN
6 1685 --          fPreferColor := NOT kPreferColor;  ( Either can't or won't )
6 1686 --          fDataHandle := Handle(GetPattern(fRsrcID));
6 1687 -2      END;
6 1688 --      FailResError;
6 1689 --      Success(fi);
6 1690 -1      END;
6 1691 --      fDefChoice := mPatternHit;
6 1692 --
6 1693 --      OffsetPtr(itsParams, SIZEOF(PatternTemplate));
6 1694 -0 A  END;
6 1695 --      {$ENDC}
6 1696 --
6 1697 --
6 1698 --      {$IFC qWriteTemplates}
6 1699 --      {$S MAWriteRes}
6 1700 --      {-----+
6 1701 --      |   WRes                                     |
6 1702 --      +-----}
6 1703 -- A  PROCEDURE TPattern.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1704 --
6 1705 --      VAR
6 1706 --          ptPtr:      PatternTemplatePtr;
6 1707 --
6 1708 0- A  BEGIN
6 1709 --      INHERITED WRes(theResource, itsParams);
6 1710 --
6 1711 --          ptPtr := PatternTemplatePtr(ExpandPtr(theResource, itsParams, SIZEOF(PatternTemplate)));
6 1712 --
6 1713 --      WITH ptPtr^ DO
6 1714 1-      BEGIN
6 1715 --          preferColor := fPreferColor;
6 1716 --      {$IFC qDebug}
6 1717 --          IF fRsrcID = kNoResource THEN
6 1718 --              WriteLn('Tried to write TPattern with no resource ID.');

```

```

6 1782 -- +-----)
6 1783 -- A  PROCEDURE TPattern.ReleasePattern;
6 1784 0- A  BEGIN
6 1785 --      fRsrcID := kNoResource;
6 1786 --      IF fPreferColor THEN
6 1787 --          DisposPixPat(PixPatHandle(fDataHandle))
6 1788 --      ELSE
6 1789 --          HPurge(fDataHandle);
6 1790 --      fDataHandle := NIL;
6 1791 -0 A  END;
6 1792 --
6 1793 --      {$S DlgNonRes}
6 1794 --      {-----+
6 1795 --      |  SetPattern
6 1796 --      +-----}
6 1797 -- A  PROCEDURE TPattern.SetPattern (thePattern: Handle; redraw: BOOLEAN);
6 1798 0- A  BEGIN
6 1799 --      ReleasePattern;
6 1800 --      fDataHandle := thePattern;
6 1801 --      IF redraw THEN
6 1802 --          ForceRedraw;
6 1803 -0 A  END;
6 1804 --
6 1805 --      {$IFC qDebug}
6 1806 --      {$S DlgFields}
6 1807 --      {-----+
6 1808 --      |  Fields
6 1809 --      +-----}
6 1810 -- A  PROCEDURE TPattern.Fields (PROCEDURE DoToField (fieldName: Str255;
6 1811 --                               fieldAddr: Ptr;
6 1812 --                               fieldType: INTEGER)); OVERRIDE;
6 1813 0- A  BEGIN
6 1814 --      DoToField('TPattern', NIL, bClass);
6 1815 --      DoToField('fPreferColor', @fPreferColor, bBoolean);
6 1816 --      DoToField('fRsrcID', @fRsrcID, bInteger);
6 1817 --      DoToField('fDataHandle', @fDataHandle, bHandle);
6 1818 --      INHERITED Fields(DoToField);
6 1819 -0 A  END;
6 1820 --
6 1821 --      {$ENDC}
6 1822 --
6 1823 --
6 1824 --      (*****
6 1825 --      (*      T P i c t u r e      *)
6 1826 --      (*****
6 1827 --
6 1828 --      {$IFC qProceduralViews}
6 1829 --      {$S DlgOpen}
6 1830 --      {-----+
6 1831 --      |  IPicture
6 1832 --      +-----}
6 1833 -- A  PROCEDURE TPicture.IPicture (itsSuperview: TView; itsLocation, itsSize: VPoint;
6 1834 --                               itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 1835 --                               itsRsrcID: INTEGER);
6 1836 --
6 1837 --      VAR
6 1838 --          fi:          FailInfo;
6 1839 --
6 1840 --      {-----+
6 1841 --      |  HandleFailure
6 1842 --      +-----}
6 1843 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1844 0- B  BEGIN
6 1845 --      Free;
6 1846 -0 B  END;
6 1847 --
6 1848 0- A  BEGIN
6 1849 --      fDataHandle := NIL;
6 1850 --      IControl(itsSuperview, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 1851 --      fRsrcID := itsRsrcID;
6 1852 --      IF fRsrcID <> kNoResource THEN
6 1853 1-  BEGIN
6 1854 --          CatchFailures(fi, HandleFailure);
6 1855 --          fDataHandle := GetPicture(fRsrcID);
6 1856 --          FailResError;
6 1857 --          Success(fi);
6 1858 -1  END;
6 1859 --      ViewEnable(FALSE, kDontRedraw);
6 1860 --      fDefChoice := mPictureHit;
6 1861 -0 A  END;
6 1862 --      {$ENDC}
6 1863 --
6 1864 --
6 1865 --      {$IFC qTemplateViews}
6 1866 --      {$S DlgOpen}
6 1867 --      {-----+
6 1868 --      |  IRes
6 1869 --      +-----}
6 1870 -- A  PROCEDURE TPicture.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
6 1871 --
6 1872 --      VAR
6 1873 --          fi:          FailInfo;
6 1874 --
6 1875 --      {-----+
6 1876 --      |  HandleFailure
6 1877 --      +-----}
6 1878 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 1879 0- B  BEGIN
6 1880 --      Free;

```



```

6 1881 -0 B      END;
6 1882 --
6 1883 0- A      BEGIN
6 1884 --          fDataHandle := NIL;
6 1885 --          INHERITED IRes(NIL, itsSuperView, itsParams);
6 1886 --
6 1887 --          fRsrcID := PictureTemplatePtr(itsParams)^.rsrcID;
6 1888 --          IF fRsrcID <> kNoResource THEN
6 1889 1-              BEGIN
6 1890 --                  CatchFailures(fi, HandleFailure);
6 1891 --                  fDataHandle := GetPicture(fRsrcID);
6 1892 --                  FailResError;
6 1893 --                  Success(fi);
6 1894 -1              END;
6 1895 --          fDefChoice := mPictureHit;
6 1896 --
6 1897 --          OffsetPtr(itsParams, SIZEOF(PictureTemplate));
6 1898 -0 A      END;
6 1899 --          {$ENDC}
6 1900 --
6 1901 --
6 1902 --          {$IFC qWriteTemplates}
6 1903 --          {$SS MAWriteRes}
6 1904 --          {-----+
6 1905 --          | WRes
6 1906 --          +-----}
6 1907 -- A      PROCEDURE TPicture.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 1908 --
6 1909 --          VAR
6 1910 --              pcPtr:      PictureTemplatePtr;
6 1911 --
6 1912 0- A      BEGIN
6 1913 --          INHERITED WRes(theResource, itsParams);
6 1914 --
6 1915 --          pcPtr := PictureTemplatePtr(ExpandPtr(theResource, itsParams, SIZEOF(PictureTemplate)));
6 1916 --
6 1917 --          {$IFC qDebug}
6 1918 --          IF fRsrcID = kNoResource THEN
6 1919 --              WriteLn('Tried to write TPicture with no resource ID.');

```

```

6 1980 0- A BEGIN
6 1981 -- fRsrcID := kNoResource;
6 1982 -- fDataHandle := NIL;
6 1983 -0 A END;
6 1984 --
6 1985 -- {$S DlgNonRes}
6 1986 -- {-----+
6 1987 -- | SetPicture |
6 1988 -- +-----}
6 1989 -- A PROCEDURE TPicture.SetPicture (thePicture: PicHandle; redraw: BOOLEAN);
6 1990 0- A BEGIN
6 1991 -- ReleasePicture;
6 1992 -- fDataHandle := thePicture;
6 1993 -- IF redraw THEN
6 1994 -- ForceRedraw;
6 1995 -0 A END;
6 1996 --
6 1997 -- {$IFC qDebug}
6 1998 -- {$S DlgFields}
6 1999 -- {-----+
6 2000 -- | Fields |
6 2001 -- +-----}
6 2002 -- A PROCEDURE TPicture.Fields (PROCEDURE DoToField (fieldName: Str255;
6 2003 -- fieldAddr: Ptr;
6 2004 -- fieldType: INTEGER)); OVERRIDE;
6 2005 0- A BEGIN
6 2006 -- DoToField('TPicture', NIL, bClass);
6 2007 -- DoToField('fRsrcID', @fRsrcID, bInteger);
6 2008 -- DoToField('fDataHandle', @fDataHandle, bHandle);
6 2009 -- INHERITED Fields(DoToField);
6 2010 -0 A END;
6 2011 --
6 2012 -- {$ENDC}
6 2013 --
6 2014 --
6 2015 -- (*****
6 2016 -- (* T P o p u p *)
6 2017 -- (*****
6 2018 --
6 2019 -- {$IFC qProceduralViews}
6 2020 -- {$S DlgOpen}
6 2021 -- {-----+
6 2022 -- | IPopUp |
6 2023 -- +-----}
6 2024 -- A PROCEDURE TPopUp.IPopUp (itsSuperview: TView; itsLocation, itsSize: VPoint;
6 2025 -- itsHSizeDet, itsVSizeDet: SizeDeterminer;
6 2026 -- itsRsrcID, itsCurrentItem, itsItemOffset: INTEGER);
6 2027 --
6 2028 -- VAR
6 2029 -- fi: FailInfo;
6 2030 -- aMenu: MenuHandle;
6 2031 --
6 2032 -- {-----+
6 2033 -- | HandleFailure |
6 2034 -- +-----}
6 2035 -- B PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 2036 0- B BEGIN
6 2037 -- Free;
6 2038 -0 B END;
6 2039 --
6 2040 0- A BEGIN
6 2041 -- fMenuHandle := NIL;
6 2042 -- IControl(itsSuperview, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 2043 --
6 2044 -- IF gConfiguration.hasHierarchicalMenus THEN
6 2045 1- BEGIN
6 2046 -- fCurrentItem := Max(1, itsCurrentItem);
6 2047 -- fItemOffset := itsItemOffset;
6 2048 -- IF itsRsrcID <> kNoResource THEN
6 2049 2- BEGIN
6 2050 -- CatchFailures(fi, HandleFailure);
6 2051 -- aMenu := GetMenu(itsRsrcID);
6 2052 -- FailNIL(aMenu);
6 2053 -- HNoPurge(Handle(aMenu));
6 2054 -- SetPopUp(aMenu, itsRsrcID, itsCurrentItem, FALSE);
6 2055 -- Success(fi);
6 2056 -2 END
6 2057 -- ELSE
6 2058 2- BEGIN
6 2059 -- fRsrcID := kNoResource;
6 2060 -- fMenuID := kNoResource;
6 2061 -2 END;
6 2062 -- fDefChoice := mPopUpHit;
6 2063 -1 END
6 2064 -- ELSE
6 2065 1- BEGIN
6 2066 -- {$IFC qDebug}
6 2067 -- ProgramBreak('Attempt to use popup menus on machine that doesn't support them');
6 2068 -- {$ENDC}
6 2069 -- fShown := FALSE; { What's reasonable here ??? }
6 2070 -1 END;
6 2071 -0 A END;
6 2072 -- {$ENDC}
6 2073 --
6 2074 --
6 2075 -- {$IFC qTemplateViews}
6 2076 -- {$S DlgOpen}
6 2077 -- {-----+
6 2078 -- | IRes |

```

```

6 2079 -- +-----}
6 2080 -- A  PROCEDURE TPopup.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
6 2081 --
6 2082 --     VAR
6 2083 --         fi:                FailInfo;
6 2084 --         aMenu:            MenuHandle;
6 2085 --
6 2086 --         {-----+
6 2087 --         |   HandleFailure   |
6 2088 --         +-----}
6 2089 -- B  PROCEDURE HandleFailure (error: OSerr; message: LONGINT);
6 2090 -- 0- B  BEGIN
6 2091 --         Free;
6 2092 -- 0 B  END;
6 2093 --
6 2094 -- 0- A  BEGIN
6 2095 --         fMenuHandle := NIL;
6 2096 --         INHERITED IRes(NIL, itsSuperview, itsParams);
6 2097 --
6 2098 --         IF gConfiguration.hasHierarchicalMenus THEN
6 2099 -- 1-         BEGIN
6 2100 --                 WITH PopupTemplatePtr(itsParams)^ DO
6 2101 -- 2-                 BEGIN
6 2102 --                         fCurrentItem := Max(1, currentItem);
6 2103 --                         fItemOffset := itemOffset;
6 2104 --                         IF rsrcID <> kNoResource THEN
6 2105 -- 3-                         BEGIN
6 2106 --                                 CatchFailures(fi, HandleFailure);
6 2107 --                                 aMenu := GetMenu(rsrcID);
6 2108 --                                 FailNIL(aMenu);
6 2109 --                                 HNoPurge(Handle(aMenu));
6 2110 --                                 SetPopup(aMenu, rsrcID, fCurrentItem, FALSE);
6 2111 --                                 Success(fi);
6 2112 -- 3-                                 END
6 2113 --                         ELSE
6 2114 -- 3-                         BEGIN
6 2115 --                                 fRsrcID := kNoResource;
6 2116 --                                 fMenuID := kNoResource;
6 2117 -- 3-                                 END;
6 2118 -- 2-                         END;
6 2119 --                         fDefChoice := mPopupHit;
6 2120 -- 1-                         END
6 2121 --                 ELSE
6 2122 -- 1-                 BEGIN
6 2123 --                         {IFC qDebug}
6 2124 --                         ProgramBreak('Attempt to use popup menus on machine that doesn't support them');
6 2125 --                         {ENDC}
6 2126 --                         fShown := FALSE;           { What's reasonable here ??? }
6 2127 -- 1-                         END;
6 2128 --
6 2129 --                         OffsetPtr(itsParams, SIZEOF(PopupTemplate));
6 2130 -- 0 A  END;
6 2131 -- {ENDC}
6 2132 --
6 2133 --
6 2134 -- {IFC qWriteTemplates}
6 2135 -- {SS MAWriteRes}
6 2136 -- {-----+
6 2137 -- |   WRes   |
6 2138 -- +-----}
6 2139 -- A  PROCEDURE TPopup.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 2140 --
6 2141 --     VAR
6 2142 --         poPtr:    PopupTemplatePtr;
6 2143 --
6 2144 -- 0- A  BEGIN
6 2145 --         INHERITED WRes(theResource, itsParams);
6 2146 --
6 2147 --         poPtr := PopupTemplatePtr(ExpandPtr(theResource, itsParams, SIZEOF(PopupTemplate)));
6 2148 --
6 2149 --         WITH poPtr^ DO
6 2150 -- 1-         BEGIN
6 2151 --         {IFC qDebug}
6 2152 --             IF fRsrcID = kNoResource THEN
6 2153 --                 ProgramBreak('Tried to write TPopup with no resource ID.');
```

```

6 2178 0- A BEGIN
6 2179 --     ReleasePopup;
6 2180 --     INHERITED Free;
6 2181 -0 A END;
6 2182 --
6 2183 --     {$S DlgOpen}
6 2184 --     {-----+
6 2185 --     | AdjustBotRight |
6 2186 --     +-----}
6 2187 -- A PROCEDURE TPopup.AdjustBotRight;
6 2188 -- VAR
6 2189 --     numItems:    INTEGER;
6 2190 --     newHeight:   INTEGER;
6 2191 --     newWidth:    INTEGER;
6 2192 0- A BEGIN
6 2193 --     IF fMenuHandle <> NIL THEN
6 2194 1- BEGIN
6 2195 --         CalcMenuSize(fMenuHandle);
6 2196 --         numItems := CountMItems(fMenuHandle);
6 2197 --         newWidth := fMenuHandle^.menuWidth + fItemOffset + finset.left + finset.right + 3;
6 2198 --         newHeight := fMenuHandle^.menuHeight DIV MAX(1, numItems) + finset.top + finset.bottom + 3;
6 2199 --         Resize(newWidth, newHeight, kRedraw);
6 2200 -1 END;
6 2201 -0 A END;
6 2202 --
6 2203 --     {$S DlgRes}
6 2204 --     {-----+
6 2205 --     | CalcLabelRect |
6 2206 --     +-----}
6 2207 -- A PROCEDURE TPopup.CalcLabelRect (VAR theRect: Rect);
6 2208 --
6 2209 0- A BEGIN
6 2210 --     ControlArea(theRect);
6 2211 --     InsetRect(theRect, 1, 1);
6 2212 --     WITH theRect DO
6 2213 1- BEGIN
6 2214 --         right := left + fItemOffset;
6 2215 --         bottom := bottom - 1;
6 2216 -1 END;
6 2217 -0 A END;
6 2218 --
6 2219 --     {$S DlgRes}
6 2220 --     {-----+
6 2221 --     | CalcMenuRect |
6 2222 --     +-----}
6 2223 -- A PROCEDURE TPopup.CalcMenuRect (VAR theRect: Rect);
6 2224 --
6 2225 0- A BEGIN
6 2226 --     ControlArea(theRect);
6 2227 --     InsetRect(theRect, 1, 1);
6 2228 --     WITH theRect DO
6 2229 1- BEGIN
6 2230 --         left := left + fItemOffset;
6 2231 --         botRight := Point(LONGINT(botRight) - $00010001);
6 2232 -1 END;
6 2233 -0 A END;
6 2234 --
6 2235 --     {$S DlgRes}
6 2236 --     {-----+
6 2237 --     | DoMouseCommand |
6 2238 --     +-----}
6 2239 -- A FUNCTION TPopup.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
6 2240 --     VAR hysteresis: Point): TCommand; OVERRIDE;
6 2241 -- VAR
6 2242 --     newChoice:    INTEGER;
6 2243 --     result:       LONGINT;
6 2244 --     menuPt:       Point;
6 2245 --     aMenuHandle:  MenuHandle;
6 2246 --     labelRect:    Rect;
6 2247 --     menuRect:     Rect;
6 2248 --     oldFColor:    RGBColor;
6 2249 --     oldBkColor:   RGBColor;
6 2250 --     newFColor:    RGBColor;
6 2251 --     newBkColor:   RGBColor;
6 2252 --
6 2253 0- A BEGIN
6 2254 --     DoMouseCommand := gNoChanges;
6 2255 --     CalcLabelRect(labelRect);
6 2256 --     CalcMenuRect(menuRect);
6 2257 --
6 2258 --     IF fMenuHandle <> NIL THEN
6 2259 1- BEGIN
6 2260 --         MAINsertMenu(fMenuHandle, hierMenu); { Use MAINsertMenu to ensure colors are set }
6 2261 --         { Save the old colors, fetch the item colors, and draw the popup box }
6 2262 --         GetIfColor(oldFColor); GetIfBkColor(oldBkColor);
6 2263 --         GetMenuColors(menuRect, fMenuID, 0, newFColor, newBkColor);
6 2264 --         SetIfColor(newBkColor); SetIfBkColor(newFColor);
6 2265 --         DrawLabel(labelRect);
6 2266 --
6 2267 --         IF gConfiguration.hasColorQD & (fRsrcID <> kNoResource) THEN
6 2268 --             aMenuHandle := GetResMenu(fRsrcID); { Reloads color tables! }
6 2269 --             SetPt(menuPt, labelRect.right, labelRect.top);
6 2270 --             LocalToGlobal(menuPt);
6 2271 --             CalcMenuSize(fMenuHandle); { Fix for Menu Manager bug }
6 2272 --             result := PopUpMenuSelect(fMenuHandle, menuPt.v, menuPt.h, fCurrentItem);
6 2273 --             newChoice := LoWord(result);
6 2274 --             SetIfColor(newFColor); SetIfBkColor(newBkColor);
6 2275 --             DrawLabel(labelRect);
6 2276 --             SetIfColor(oldFColor); SetIfBkColor(oldBkColor);

```

```

6 2277 --      IF HiWord(result) <> 0 THEN
6 2278 2-      BEGIN
6 2279 --          SetCurrentItem(newChoice, TRUE);
6 2280 --          DoChoice(SELf, fDefChoice);
6 2281 -2      END;
6 2282 --      DeleteMenu(fMenuID);
6 2283 --      SetIfColor(oldFColor); SetIfBkColor(oldBkColor);
6 2284 -1      END;
6 2285 -0 A  END;
6 2286 --
6 2287 --      {$S DlgRes}
6 2288 --      {-----+
6 2289 --      | Draw                                     |
6 2290 --      +-----+}
6 2291 -- A  PROCEDURE TPopup.Draw (area: Rect); OVERRIDE;
6 2292 --
6 2293 --      VAR
6 2294 --          aRect:      Rect;
6 2295 --          oldFColor:   RGBColor;
6 2296 --          oldBkColor:  RGBColor;
6 2297 --          newFColor:   RGBColor;
6 2298 --          newBkColor:  RGBColor;
6 2299 --
6 2300 0- A  BEGIN
6 2301 --      IF fMenuHandle <> NIL THEN
6 2302 1-      BEGIN
6 2303 --          MAINsertMenu(fMenuHandle, hierMenu);    { Use MAINsertMenu to ensure colors are set }
6 2304 --          { Erase the whole menu first }
6 2305 --          ControlArea(aRect);
6 2306 --          IF SectRect(area, aRect, aRect) THEN
6 2307 2-      BEGIN
6 2308 --          (* EraseRect(aRect); *)
6 2309 --
6 2310 --          { Save the old colors, fetch the item colors, and draw the popup box }
6 2311 --          GetIfColor(oldFColor); GetIfBkColor(oldBkColor);
6 2312 --          CalcMenuRect(aRect);
6 2313 --          GetMenuColors(aRect, fMenuID, fCurrentItem, newFColor, newBkColor);
6 2314 --          SetIfColor(newFColor); SetIfBkColor(newBkColor);
6 2315 --          DrawPopupBox(area);
6 2316 --
6 2317 --          { Fetch the title colors, and draw it }
6 2318 --          GetMenuColors(aRect, fMenuID, 0, newFColor, newBkColor);
6 2319 --          SetIfColor(newFColor); SetIfBkColor(newBkColor);
6 2320 --          DrawLabel(area);
6 2321 --
6 2322 --          { Reset colors to their original state }
6 2323 --          SetIfColor(oldFColor); SetIfBkColor(oldBkColor);
6 2324 -2      END;
6 2325 --      DeleteMenu(fMenuID);
6 2326 -1      END;
6 2327 --
6 2328 --      INHERITED Draw(area);
6 2329 -0 A  END;
6 2330 --
6 2331 --      {$S DlgRes}
6 2332 --      {-----+
6 2333 --      | DrawLabel                               |
6 2334 --      +-----+}
6 2335 -- A  PROCEDURE TPopup.DrawLabel (area: Rect);
6 2336 --
6 2337 --      CONST
6 2338 --          labelSlop      = 5;
6 2339 --
6 2340 --      VAR
6 2341 --          labelRect:      Rect;
6 2342 --          theLabel:       Str255;
6 2343 --
6 2344 0- A  BEGIN
6 2345 --          CalcLabelRect(labelRect);
6 2346 --          labelRect.right := labelRect.right - labelSlop;
6 2347 --          IF SectRect(area, labelRect, area) THEN
6 2348 1-      BEGIN
6 2349 --          theLabel := CONCAT(fMenuHandle^.menuData, ':'); { Fetch the title of the menu }
6 2350 --          TextBox(Ptr(ORD4(@theLabel)+1), LENGTH(theLabel), labelRect, teJustRight);
6 2351 -1      END;
6 2352 -0 A  END;
6 2353 --
6 2354 --      {$S DlgRes}
6 2355 --      {-----+
6 2356 --      | DrawPopupBox                             |
6 2357 --      +-----+}
6 2358 -- A  PROCEDURE TPopup.DrawPopupBox (area: Rect);
6 2359 --
6 2360 --      CONST
6 2361 --          ShadowedFrame = [adnLineTop, adnLineLeft, adnLineBottom, adnLineRight, adnShadow];
6 2362 --          leftSlop      = 12;
6 2363 --          rightSlop     = 0;
6 2364 --          botSlop       = 4;
6 2365 --
6 2366 --      VAR
6 2367 --          wid:           INTEGER;
6 2368 --          newWid:        INTEGER;
6 2369 --          newLen:        INTEGER;
6 2370 --          menuRect:      Rect;
6 2371 --          colorRect:     Rect;
6 2372 --          theLabel:      Str255;
6 2373 --
6 2374 0- A  BEGIN
6 2375 --          CalcMenuRect(menuRect);

```

```

6 2376 --      GetItem(fMenuHandle, fCurrentItem, theLabel);
6 2377 --      WITH menuRect DO
6 2378 1-      BEGIN
6 2379 --          IF SectRect(area, menuRect, colorRect) THEN
6 2380 2-      BEGIN
6 2381 --          wid := (right - left) - (leftSlop + rightSlop);
6 2382 --          newWid := StringWidth(theLabel);
6 2383 --          IF newWid > wid THEN
6 2384 3-      BEGIN
6 2385 --          newLen := LENGTH(theLabel);
6 2386 --          wid := wid - CharWidth('_');
6 2387 --
6 2388 4-      REPEAT
6 2389 --          newWid := newWid - CharWidth(theLabel[newLen]);
6 2390 --          newLen := PRED(newLen);
6 2391 --          UNTIL (newWid <= wid) OR (LENGTH(theLabel) = 0);
6 2392 --
6 2393 --          newLen := SUCC(newLen);
6 2394 --          theLabel[newLen] := '_';
6 2395 --          theLabel[0] := CHR(newLen);
6 2396 -3      END;
6 2397 --
6 2398 --      PenNormal;
6 2399 --
6 2400 --      EraseRect(colorRect);
6 2401 --
6 2402 --      MoveTo(left + leftSlop, bottom - botSlop);
6 2403 --      DrawString(theLabel);
6 2404 --      SetIfColor(gRGBBlack);
6 2405 --      InsetRect(menuRect, -1, -1);
6 2406 --      botRight := Point(LONGINT(botRight) - $00010001);
6 2407 --      FrameRect(menuRect);
6 2408 --      MoveTo(left + 2, bottom);
6 2409 --      LineTo(right, bottom);
6 2410 --      LineTo(right, top + 2);
6 2411 -2      END;
6 2412 -1      END;
6 2413 -0 A  END;
6 2414 --
6 2415 --      {$$ DlgNonRes}
6 2416 --      {-----+
6 2417 --      |  GetCurrentItem      |
6 2418 --      +-----+}
6 2419 -- A  FUNCTION TPopup.GetCurrentItem: INTEGER;
6 2420 0- A  BEGIN
6 2421 --      GetCurrentItem := fCurrentItem;
6 2422 -0 A  END;
6 2423 --
6 2424 --      {$$ DlgNonRes}
6 2425 --      {-----+
6 2426 --      |  GetItemText        |
6 2427 --      +-----+}
6 2428 -- A  PROCEDURE TPopup.GetItemText (item: INTEGER; VAR theText: Str255);
6 2429 0- A  BEGIN
6 2430 --      IF fMenuHandle <> NIL THEN
6 2431 --          GetItem(fMenuHandle, item, theText)
6 2432 --      ELSE
6 2433 --          theText := '';
6 2434 -0 A  END;
6 2435 --
6 2436 --      {$$ DlgNonRes}
6 2437 --      {-----+
6 2438 --      |  ReleasePopup      |
6 2439 --      +-----+}
6 2440 -- A  PROCEDURE TPopup.ReleasePopup;
6 2441 0- A  BEGIN
6 2442 --      IF fMenuHandle <> NIL THEN
6 2443 1-      BEGIN
6 2444 --          ($IFC FALSE)
6 2445 --          DeleteMenu(fMenuID);
6 2446 --          ($ENDC)
6 2447 --          HPurge(Handle(fMenuHandle));
6 2448 --          DisposHandle(Handle(fMenuHandle));
6 2449 --          fMenuHandle := NIL;
6 2450 -1      END;
6 2451 --      fRsrcID := kNoResource;
6 2452 --      fMenuID := kNoResource;
6 2453 --      fCurrentItem := 0;
6 2454 -0 A  END;
6 2455 --
6 2456 --      {$$ DlgNonRes}
6 2457 --      {-----+
6 2458 --      |  SetCurrentItem    |
6 2459 --      +-----+}
6 2460 -- A  PROCEDURE TPopup.SetCurrentItem (item: INTEGER; redraw: BOOLEAN);
6 2461 --      VAR
6 2462 --          menuRect:      Rect;
6 2463 0- A  BEGIN
6 2464 --      IF (fMenuHandle <> NIL) AND (item <> fCurrentItem) THEN
6 2465 1-      BEGIN
6 2466 --          IF fCurrentItem <> 0 THEN
6 2467 --              SetItemMark(fMenuHandle, fCurrentItem, ' ');
6 2468 --          IF item <> 0 THEN
6 2469 --              SetItemMark(fMenuHandle, item, CHR(checkMark));
6 2470 --          fCurrentItem := item;
6 2471 -1      END;
6 2472 --      IF redraw THEN
6 2473 1-      BEGIN
6 2474 --          CalcMenuRect(menuRect);

```

```

6 2475 --      menuRect.botRight := Point(LONGINT(menuRect.botRight) - $00010001);
6 2476 --      InvalidRect(menuRect);
6 2477 -1      END;
6 2478 -0 A  END;
6 2479 --
6 2480 --      {$S DlgRes}
6 2481 --      {-----+      theMenu is assumed to be a handle to a real MENU. If it
6 2482 --      | SetPopup      |      is from a resource, the handle will be detached!
6 2483 --      +-----+}
6 2484 -- A  PROCEDURE TPopup.SetPopup (theMenu: MenuHandle; theRsrcID, currentItem: INTEGER; redraw: BOOLEAN);
6 2485 --
6 2486 --      VAR
6 2487 --          fi:      FailInfo;
6 2488 --          aMenu:   MenuHandle;
6 2489 --
6 2490 --      {-----+
6 2491 --      | HandleFailure      |
6 2492 --      +-----+}
6 2493 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 2494 0- B  BEGIN
6 2495 --      Free;
6 2496 -0 B  END;
6 2497 --
6 2498 0- A  BEGIN
6 2499 --      ReleasePopup;
6 2500 --      IF theMenu <> NIL THEN
6 2501 1-      BEGIN
6 2502 --          IF theRsrcID <> kNoResource THEN
6 2503 2-      BEGIN
6 2504 --          CatchFailures(fi, HandleFailure);
6 2505 --          DetachResource(Handle(theMenu));
6 2506 --          FailResError;
6 2507 --          Success(fi);
6 2508 -2      END;
6 2509 --          {$IFC FALSE}
6 2510 --          MAInsertMenu(theMenu, hierMenu);      { Use MAInsertMenu to ensure colors are set }
6 2511 --          {$ENDC}
6 2512 --          fMenuHandle := theMenu;
6 2513 --          fRsrcID := theRsrcID;
6 2514 --          fMenuID := theMenu^.menuID;
6 2515 -1      END;
6 2516 --          SetCurrentItem(MAX(1, currentItem), kDontRedraw);
6 2517 --          AdjustBotRight;
6 2518 --          IF redraw THEN
6 2519 --              ForceRedraw;
6 2520 -0 A  END;
6 2521 --
6 2522 --      {$IFC qDebug}
6 2523 --      {$S DlgFields}
6 2524 --      {-----+
6 2525 --      | Fields      |
6 2526 --      +-----+}
6 2527 -- A  PROCEDURE TPopup.Fields (PROCEDURE DoToField (fieldName: Str255;
6 2528 --                                fieldAddr: Ptr;
6 2529 --                                fieldType: INTEGER)); OVERRIDE;
6 2530 0- A  BEGIN
6 2531 --      DoToField('TPopup', NIL, bClass);
6 2532 --      DoToField('fRsrcID', @fRsrcID, bInteger);
6 2533 --      DoToField('fMenuID', @fMenuID, bInteger);
6 2534 --      DoToField('fMenuHandle', @fMenuHandle, bHandle);
6 2535 --      DoToField('fCurrentItem', @fCurrentItem, bInteger);
6 2536 --      DoToField('fItemOffset', @fItemOffset, bInteger);
6 2537 --      INHERITED Fields(DoToField);
6 2538 -0 A  END;
6 2539 --
6 2540 --      {$ENDC}
6 2541 --
6 2542 --
6 2543 --      (*****
6 2544 --      (*      T D i a l o g T E V i e w      *)
6 2545 --      (*****
6 2546 --      {$S DlgRes}
6 2547 --      {-----+
6 2548 --      | DrawContents      |
6 2549 --      +-----+}
6 2550 -- A  PROCEDURE TDialogTEView.DrawContents; OVERRIDE;
6 2551 0- A  BEGIN
6 2552 --      { Rely on the TEditText view to draw this }
6 2553 -0 A  END;
6 2554 --
6 2555 --
6 2556 --      {$IFC qDebug}
6 2557 --      {$S DlgFields}
6 2558 --      {-----+
6 2559 --      | Fields      |
6 2560 --      +-----+}
6 2561 -- A  PROCEDURE TDialogTEView.Fields (PROCEDURE DoToField (fieldName: Str255;
6 2562 --                                fieldAddr: Ptr;
6 2563 --                                fieldType: INTEGER)); OVERRIDE;
6 2564 0- A  BEGIN
6 2565 --      DoToField('TDialogTEView', NIL, bClass);
6 2566 --      DoToField('fEditText', @fEditText, bObject);
6 2567 --      INHERITED Fields(DoToField);
6 2568 -0 A  END;
6 2569 --      {$ENDC}
6 2570 --
6 2571 --
6 2572 --      {$S DlgNonRes}
6 2573 --      {-----+

```

```

6 2574 -- | InstallEditText
6 2575 -- +-----+
6 2576 -- A PROCEDURE TDialogTEView.InstallEditText (theEditText: TEditText; selectChars: BOOLEAN);
6 2577 --
6 2578 -- VAR
6 2579 --     theText:      Str255;
6 2580 --
6 2581 0- A BEGIN
6 2582 --     SetJustification(theEditText.fJust, kDontRedraw);
6 2583 --
6 2584 --     IF fSuperView <> NIL THEN
6 2585 --         fSuperView.RemoveSubview(SELF);
6 2586 --         theEditText.fSuperView.AddSubview(SELF);
6 2587 --
6 2588 --         fInset := theEditText.fInset;
6 2589 --         SetTextStyle(0, 0, doAll, theEditText.fTextStyle, kDontRedraw);
6 2590 --         {$H-}
6 2591 --         WITH theEditText.fSize DO
6 2592 --             Resize(h, Max(v, fHTE^.lineHeight + fInset.top + fInset.bottom), kDontRedraw);
6 2593 --         WITH theEditText.fLocation DO
6 2594 --             Locate(h, v, kDontRedraw);
6 2595 --         {$H+}
6 2596 --
6 2597 --         theEditText.GetText(theText);
6 2598 --         SetText(theText);
6 2599 --         RecalcText;
6 2600 --         IF selectChars THEN
6 2601 --             SetSelect(0, MAXINT, fHTE);
6 2602 --         ELSE
6 2603 --             SetSelect(0, 0, fHTE);           { Caller will set the selection. }
6 2604 --
6 2605 --         fMaxChars := theEditText.fMaxChars;
6 2606 --         fEditText := theEditText;
6 2607 --         theEditText.fTEView := SELF;
6 2608 --         theEditText.fIdleFreq := fIdleFreq;
6 2609 0- A END;
6 2610 --
6 2611 --
6 2612 -- {$S DlgNonRes}
6 2613 -- {-----+
6 2614 -- | InstallSelection
6 2615 -- +-----+}
6 2616 -- A PROCEDURE TDialogTEView.InstallSelection(wasActive, beActive: BOOLEAN); OVERRIDE;
6 2617 --
6 2618 0- A BEGIN
6 2619 --     { If we're deselecting a field and it's been horizontally scrolled, invalidate it
6 2620 --       so that it is redrawn correctly.}
6 2621 --     IF NOT beActive THEN
6 2622 --         IF fHTE^.destRect.left <> fInset.left THEN
6 2623 1-             BEGIN
6 2624 --                 InvalidateRect(fHTE^.viewRect);
6 2625 --                 fHTE^.destRect.left := fInset.left;
6 2626 -1             END;
6 2627 --
6 2628 --     INHERITED InstallSelection(wasActive, beActive);
6 2629 0- A END;
6 2630 --
6 2631 --
6 2632 -- {$S DlgRes}
6 2633 -- {-----+
6 2634 -- | RecalcText
6 2635 -- +-----+}
6 2636 -- A PROCEDURE TDialogTEView.RecalcText; OVERRIDE;
6 2637 0- A BEGIN
6 2638 --     INHERITED RecalcText;
6 2639 --     IF fHTE^.just = teJustLeft THEN           { This is awful kludgy. Any better determiner? }
6 2640 --         fHTE^.destRect.right := kMaxTEwidth;
6 2641 0- A END;
6 2642 --
6 2643 --
6 2644 -- {$S DlgRes}
6 2645 -- {-----+
6 2646 -- | ScrollSelectionIntoView
6 2647 -- +-----+}
6 2648 -- A PROCEDURE TDialogTEView.ScrollSelectionIntoView; OVERRIDE;
6 2649 --
6 2650 0- A BEGIN
6 2651 --     {$IFC NOT qNeedsROM128K}
6 2652 --         IF qConfiguration.hasROM128K THEN
6 2653 --             {$ENDC}
6 2654 --             IF Focus THEN
6 2655 --                 TESelView(fHTE);           { This assumes, of course, NOT fAutoWrap... }
6 2656 0- A END;
6 2657 --
6 2658 --
6 2659 -- (*-----*)
6 2660 -- (* TStaticText *)
6 2661 -- (*-----*)
6 2662 --
6 2663 -- {$IFC qProceduralViews}
6 2664 -- {$S DlgOpen}
6 2665 -- {-----+
6 2666 -- | IStaticText
6 2667 -- +-----+}
6 2668 -- A PROCEDURE TStaticText.IStaticText (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 2669 --                                     itsHSizedet, itsVSizedet: SizeDeterminer;
6 2670 --                                     itsRsrcID, itsIndex: INTEGER);
6 2671 --
6 2672 -- VAR
6 2673 --     astring:      Str255;

```



```

6 2673 --      fi:          FailInfo;
6 2674 --
6 2675 --      {-----+
6 2676 --      |   HandleFailure
6 2677 --      +-----}
6 2678 --      PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
6 2679 0- B      BEGIN
6 2680 --          Free;
6 2681 -0 B      END;
6 2682 --
6 2683 0- A      BEGIN
6 2684 --          fDataHandle := NIL;
6 2685 --          IControl(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
6 2686 --          fRsrcID := itsRsrcID;
6 2687 --          fIndex := itsIndex;
6 2688 --          fJust := teJustLeft;          { Default to system justification }
6 2689 --          IF fRsrcID <> kNoResource THEN
6 2690 1-              BEGIN
6 2691 --                  CatchFailures(fi, HandleFailure);
6 2692 --                  GetIndString(aString, fRsrcID, fIndex);
6 2693 --                  FailResError;
6 2694 --                  Success(fi);
6 2695 --                  SetText(aString, kDontRedraw);
6 2696 -1                  END;
6 2697 --                  ViewEnable(FALSE, kDontRedraw);          { Default is to not enable hit testing }
6 2698 --                  fDefChoice := mStaticTextHit;
6 2699 -0 A      END;
6 2700 --      {$ENDC}
6 2701 --
6 2702 --
6 2703 --      {$IFC qTemplateViews}
6 2704 --      {$S DlgOpen}
6 2705 --      {-----+
6 2706 --      |   IRes
6 2707 --      +-----}
6 2708 --      A      PROCEDURE TStaticText.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 2709 --
6 2710 0- A      BEGIN
6 2711 --          fDataHandle := NIL;
6 2712 --          INHERITED IRes(NIL, itsSuperView, itsParams);
6 2713 --
6 2714 --          fDefChoice := mStaticTextHit;
6 2715 --          WITH StaticTextTemplatePtr(itsParams)^ DO
6 2716 1-              BEGIN
6 2717 --                  fJust := just;
6 2718 --                  SetText(data, kDontRedraw);
6 2719 -1                  END;
6 2720 --
6 2721 --          OffsetPtrWStr(itsParams, SIZEOF(StaticTextTemplate));
6 2722 -0 A      END;
6 2723 --      {$ENDC}
6 2724 --
6 2725 --
6 2726 --      {$IFC qWriteTemplates}
6 2727 --      {$S MAWriteRes}
6 2728 --      {-----+
6 2729 --      |   WRes
6 2730 --      +-----}
6 2731 --      A      PROCEDURE TStaticText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 2732 --
6 2733 --      VAR
6 2734 --          theText:   Str255;
6 2735 --          stPtr:     StaticTextTemplatePtr;
6 2736 --
6 2737 0- A      BEGIN
6 2738 --          INHERITED WRes(theResource, itsParams);
6 2739 --
6 2740 --          GetText(theText);
6 2741 --
6 2742 --          stPtr := StaticTextTemplatePtr(ExpandPtrWStr(theResource, itsParams,
6 2743 --              SIZEOF(StaticTextTemplate), LENGTH(theText)));
6 2744 --
6 2745 --          WITH stPtr^ DO
6 2746 1-              BEGIN
6 2747 --                  just := fJust;
6 2748 --                  (* data := theText; *)
6 2749 --                  CopyStr255(theText, PRStr(data));
6 2750 -1                  END;
6 2751 -0 A      END;
6 2752 --
6 2753 --      {$S MAWriteRes}
6 2754 --      {-----+
6 2755 --      |   WriteRes
6 2756 --      +-----}
6 2757 --      A      PROCEDURE TStaticText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 2758 0- A      BEGIN
6 2759 --          gWResSignature := 'stat';   gWResType := 'TStaticText';
6 2760 --          WRes(theResource, itsParams);
6 2761 -0 A      END;
6 2762 --      {$ENDC}
6 2763 --
6 2764 --
6 2765 --      {$S DlgClose}
6 2766 --      {-----+
6 2767 --      |   Free
6 2768 --      +-----}
6 2769 --      A      PROCEDURE TStaticText.Free; OVERRIDE;
6 2770 0- A      BEGIN
6 2771 --          ReleaseText;

```

```

6 2772 --      INHERITED Free;
6 2773 -0 A  END;
6 2774 --
6 2775 --      {$S DlgRes}
6 2776 --      {-----+
6 2777 --      | DoSubstitution
6 2778 --      +-----}
6 2779 -- A  FUNCTION TStaticText.DoSubstitution (VAR realText: Handle): BOOLEAN;
6 2780 --
6 2781 --      VAR
6 2782 --          err:          OSerr;
6 2783 --          aDialogView:  TDialogView;
6 2784 --
6 2785 0- A  BEGIN
6 2786 --          realText := fDataHandle;          { Make a copy of the original text }
6 2787 --          err := HandToHand(realText);
6 2788 --          IF err <> noErr THEN                { We just won't do substitution in an emergency }
6 2789 --              realText := fDataHandle
6 2790 --          ELSE
6 2791 1-      BEGIN
6 2792 --          aDialogView := TDialogView(GetDialogView);
6 2793 --          IF aDialogView <> NIL THEN
6 2794 --              aDialogView.ReplaceText(realText);
6 2795 1-      END;
6 2796 --          DoSubstitution := (err = noErr);
6 2797 -0 A  END;
6 2798 --
6 2799 --      {$S DlgRes}
6 2800 --      {-----+
6 2801 --      | Draw
6 2802 --      +-----}
6 2803 -- A  PROCEDURE TStaticText.Draw (area: Rect); OVERRIDE;
6 2804 --      VAR
6 2805 --          subbed:      BOOLEAN;
6 2806 --          theRect:     Rect;
6 2807 --          oldColor:    RGBColor;
6 2808 --          realText:    Handle;
6 2809 --
6 2810 0- A  BEGIN
6 2811 --      IF fDataHandle <> NIL THEN
6 2812 1-      BEGIN
6 2813 --          subbed := DoSubstitution(realText); { Make the substitution if desired }
6 2814 --          ControlArea(theRect);
6 2815 --          PenNormal; { ??? NECESSARY ??? }
6 2816 --          GetIfColor(oldColor);
6 2817 --          SetPortTextStyle(fTextStyle);
6 2818 --          HLock(realText);
6 2819 --          ImageText(Ptr(ORD4(realText^) + 1), { Draw the text itself }
6 2820 --              LENGTH(StringHandle(realText)^), theRect, fJust);
6 2821 --          HUnlock(realText);
6 2822 --          IF subbed THEN
6 2823 --              DisposHandle(realText);
6 2824 --          SetIfColor(oldColor);
6 2825 1-      END;
6 2826 --      INHERITED Draw(area);
6 2827 -0 A  END;
6 2828 --
6 2829 --      {$S DlgRes}
6 2830 --      {-----+
6 2831 --      | GetText
6 2832 --      +-----}
6 2833 -- A  PROCEDURE TStaticText.GetText (VAR theText: Str255);
6 2834 --
6 2835 0- A  BEGIN
6 2836 --      IF fDataHandle <> NIL THEN
6 2837 --          theText := StringHandle(fDataHandle)^
6 2838 --      ELSE
6 2839 --          theText := '';
6 2840 -0 A  END;
6 2841 --
6 2842 --      {$S DlgRes}
6 2843 --      {-----+
6 2844 --      | ImageText
6 2845 --      +-----}
6 2846 -- A  PROCEDURE TStaticText.ImageText (text: Ptr; length: LONGINT; box: Rect; just: INTEGER);
6 2847 0- A  BEGIN
6 2848 --      TextBox(text, length, box, just);
6 2849 -0 A  END;
6 2850 --
6 2851 --      {$S DlgNonRes}
6 2852 --      {-----+
6 2853 --      | ReleaseText
6 2854 --      +-----}
6 2855 -- A  PROCEDURE TStaticText.ReleaseText;
6 2856 0- A  BEGIN
6 2857 --      DisposIfHandle(fDataHandle);
6 2858 --      fRsrcID := kNoResource;
6 2859 --      fDataHandle := NIL;
6 2860 -0 A  END;
6 2861 --
6 2862 --      {$S DlgNonRes}
6 2863 --      {-----+
6 2864 --      | SetJustification
6 2865 --      +-----}
6 2866 -- A  PROCEDURE TStaticText.SetJustification (theJust: INTEGER; redraw: BOOLEAN);
6 2867 0- A  BEGIN
6 2868 --      fJust := theJust;
6 2869 --      IF redraw THEN
6 2870 --          ForceRedraw;

```

```

6 2871 -0 A  END;
6 2872 --
6 2873 --      {$S DlgNonRes}
6 2874 --      {-----+
6 2875 --      |   SetText           |
6 2876 --      +-----}
6 2877 -- A  PROCEDURE TStaticText.SetText (theText: Str255; redraw: BOOLEAN);
6 2878 --      VAR
6 2879 --          area:      Rect;
6 2880 --
6 2881 0- A  BEGIN
6 2882 --      IF (fDataHandle = NIL) | (theText <> StringHandle(fDataHandle)^^) THEN
6 2883 1-          BEGIN
6 2884 --              ReleaseText;
6 2885 --              fDataHandle := Handle(NewString(theText));
6 2886 --              IF MemError <> noErr THEN
6 2887 --                  fDataHandle := NIL;
6 2888 --              IF redraw & Focus THEN
6 2889 2-                  BEGIN
6 2890 --                      ControlArea(area);
6 2891 --                      EraseRect(area);
6 2892 --                      Draw(area);
6 2893 -2                      END;
6 2894 -1                      END;
6 2895 -0 A  END;
6 2896 --
6 2897 --      {$IFC qDebug}
6 2898 --      {$S DlgFields}
6 2899 --      {-----+
6 2900 --      |   Fields           |
6 2901 --      +-----}
6 2902 -- A  PROCEDURE TStaticText.Fields (PROCEDURE DoToField (fieldName: Str255;
6 2903 --                                fieldAddr: Ptr;
6 2904 --                                fieldType: INTEGER)); OVERRIDE;
6 2905 0- A  BEGIN
6 2906 --      DoToField('TStaticText', NIL, bClass);
6 2907 --      DoToField('fRsrcID', @fRsrcID, bInteger);
6 2908 --      DoToField('fIndex', @fIndex, bInteger);
6 2909 --      DoToField('fDataHandle', @fDataHandle, bHandle);
6 2910 --      DoToField('fJust', @fJust, bInteger);
6 2911 --      INHERITED Fields(DoToField);
6 2912 -0 A  END;
6 2913 --      {$ENDC}
6 2914 --
6 2915 --
6 2916 --      (*****
6 2917 --      (*      T E d i t T e x t      *)
6 2918 --      (*****
6 2919 --
6 2920 --      {$IFC qProceduralViews}
6 2921 --      {$S DlgOpen}
6 2922 --      {-----+
6 2923 --      |   IEditText           |
6 2924 --      +-----}
6 2925 -- A  PROCEDURE TEditText.IEditText (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 2926 --                                itsMaxChars: INTEGER);
6 2927 --      VAR
6 2928 --          itsMargins:      Rect;
6 2929 --
6 2930 0- A  BEGIN
6 2931 --      fTEView := NIL;
6 2932 --      IStaticText(itsSuperView, itsLocation, itsSize, sizeFixed, sizeFixed,
6 2933 --                  kNoResource, 0);
6 2934 --      fMaxChars := itsMaxChars;
6 2935 --      fControlChars := [chLeft, chRight, chUp, chDown, chBackspace];
6 2936 --      fTextStyle := gSystemStyle;
6 2937 --      Inset(2, 2, kDontRedraw);           { Default is a little, teeny inset... }
6 2938 --      fPenSize := Point($00010001);      { ...and a thin frame }
6 2939 --      fAdornment := kFrame;
6 2940 --      ViewEnable(TRUE, kDontRedraw);
6 2941 --      fDefChoice := mEditTextHit;
6 2942 -0 A  END;
6 2943 --      {$ENDC}
6 2944 --
6 2945 --
6 2946 --      {$IFC qTemplateViews}
6 2947 --      {$S DlgOpen}
6 2948 --      {-----+
6 2949 --      |   IRes                 |
6 2950 --      +-----}
6 2951 -- A  PROCEDURE TEditText.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 2952 --
6 2953 0- A  BEGIN
6 2954 --      fTEView := NIL;
6 2955 --      INHERITED IRes(NIL, itsSuperView, itsParams);
6 2956 --
6 2957 --      WITH EditTextTemplatePtr(itsParams)^ DO
6 2958 1-          BEGIN
6 2959 --              fMaxChars := maxChars;
6 2960 --              fControlChars := controlChars;
6 2961 -1          END;
6 2962 --      fDefChoice := mEditTextHit;
6 2963 --
6 2964 --      OffsetPtr(itsParams, SIZEOF(EditTextTemplate));
6 2965 -0 A  END;
6 2966 --      {$ENDC}
6 2967 --
6 2968 --
6 2969 --      {$IFC qWriteTemplates}

```

```

6 2970 --      {$$ MAWriteRes}
6 2971 --      {-----+
6 2972 --      | WRes
6 2973 --      +-----}
6 2974 -- A  PROCEDURE TEditText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 2975 --
6 2976 --      VAR
6 2977 --          edPtr:      EditTextTemplatePtr;
6 2978 --
6 2979 -- A  BEGIN
6 2980 --      INHERITED WRes(theResource, itsParams);
6 2981 --
6 2982 --      edPtr := EditTextTemplatePtr(ExpandPtr(theResource, itsParams, SIZEOF(EditTextTemplate)));
6 2983 --
6 2984 --      WITH edPtr^ DO
6 2985 --      BEGIN
6 2986 --          maxChars := fmaxChars;
6 2987 --          controlChars := fControlChars;
6 2988 --      END;
6 2989 -- A  END;
6 2990 --
6 2991 --      {$$ MAWriteRes}
6 2992 --      {-----+
6 2993 --      | WriteRes
6 2994 --      +-----}
6 2995 -- A  PROCEDURE TEditText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 2996 -- A  BEGIN
6 2997 --      gWResSignature := 'edit';  gWResType := 'TEditText';
6 2998 --      WRes(theResource, itsParams);
6 2999 -- A  END;
6 3000 --      {$ENDC}
6 3001 --
6 3002 --
6 3003 --      {$$ DlgRes}
6 3004 --      {-----+
6 3005 --      | DoIdle
6 3006 --      +-----}
6 3007 -- A  PROCEDURE TEditText.DoIdle (phase: IdlePhase); OVERRIDE;
6 3008 --
6 3009 -- A  BEGIN
6 3010 --      IF fTextView <> NIL THEN
6 3011 --          fTextView.DoIdle(phase);
6 3012 -- A  END;
6 3013 --
6 3014 --      {$$ DlgRes}
6 3015 --      {-----+
6 3016 --      | DoKeyCommand
6 3017 --      +-----}
6 3018 -- A  FUNCTION TEditText.DoKeyCommand (ch: CHAR; aKeyCode: INTEGER; VAR info: EventInfo): TCommand; OVERRIDE;
6 3019 --
6 3020 --      VAR
6 3021 --          aCommand:      TCommand;
6 3022 --
6 3023 -- A  BEGIN
6 3024 --      IF (fTextView <> NIL) & ((ch >= ' ') | (ch IN fControlChars)) THEN
6 3025 --      BEGIN
6 3026 --          aCommand := fTextView.DoKeyCommand(ch, aKeyCode, info);
6 3027 --          IF aCommand <> gNoChanges THEN
6 3028 --              aCommand.fCanUndo := FALSE;      { !!! Temporary until commands have targets}
6 3029 --          DoKeyCommand := aCommand;
6 3030 --      END
6 3031 --      ELSE
6 3032 --          DoKeyCommand := INHERITED DoKeyCommand(ch, aKeyCode, info);
6 3033 -- A  END;
6 3034 --
6 3035 --      {$$ DlgRes}
6 3036 --      {-----+
6 3037 --      | DoMenuCommand
6 3038 --      +-----}
6 3039 -- A  FUNCTION TEditText.DoMenuCommand (aCmdNumber: CmdNumber): TCommand; OVERRIDE;
6 3040 --
6 3041 --      VAR
6 3042 --          aCommand:      TCommand;
6 3043 --
6 3044 -- A  BEGIN
6 3045 --      IF fTextView <> NIL THEN
6 3046 --      BEGIN
6 3047 --          aCommand := fTextView.DoMenuCommand(aCmdNumber);
6 3048 --          IF aCommand <> gNoChanges THEN
6 3049 --              aCommand.fCanUndo := FALSE;      { !!! Temporary until commands have targets}
6 3050 --          DoMenuCommand := aCommand;
6 3051 --      END
6 3052 --      ELSE
6 3053 --          DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
6 3054 -- A  END;
6 3055 --
6 3056 --
6 3057 --      {$$ DlgRes}
6 3058 --      {-----+
6 3059 --      | DoMouseCommand
6 3060 --      +-----}
6 3061 -- A  FUNCTION TEditText.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
6 3062 --      VAR hysteresis: Point): TCommand; OVERRIDE;
6 3063 --
6 3064 --      VAR
6 3065 --          aCommand:      TCommand;
6 3066 --
6 3067 -- A  BEGIN
6 3068 --      IF gTarget <> SELF THEN

```

```

6 3069 --      DoChoice(SELF, mEditTextHit);
6 3070 --
6 3071 --      { Make sure we were in fact selected before passing along the mouse click }
6 3072 --      IF qTarget = SELF THEN
6 3073 1-      BEGIN
6 3074 --          aCommand := fTextView.DoMouseCommand(theMouse, info, hysteresis);
6 3075 --          IF aCommand <> gNoChanges THEN
6 3076 --              aCommand.fCanUndo := FALSE;      { !!! Temporary until commands have targets }
6 3077 --          DoMouseCommand := aCommand;
6 3078 1-      END
6 3079 --      ELSE
6 3080 --          DoMouseCommand := gNoChanges;
6 3081 0- A  END;
6 3082 --
6 3083 --
6 3084 --      {$S DlgRes}
6 3085 --      {-----+
6 3086 --      | DoSetupMenus
6 3087 --      +-----}
6 3088 -- A  PROCEDURE TEditText.DoSetupMenus; OVERRIDE;
6 3089 --
6 3090 0- A  BEGIN
6 3091 --      IF fTextView <> NIL THEN
6 3092 --          fTextView.DoSetupMenus
6 3093 --      ELSE
6 3094 --          INHERITED DoSetupMenus;
6 3095 0- A  END;
6 3096 --
6 3097 --
6 3098 --      {$S DlgRes}
6 3099 --      {-----+
6 3100 --      | DoSubstitution
6 3101 --      +-----}
6 3102 -- A  FUNCTION TEditText.DoSubstitution (VAR realText: Handle): BOOLEAN; OVERRIDE;
6 3103 0- A  BEGIN
6 3104 --      { Default action is for editable text items is not to do any substitutions }
6 3105 --      realText := fDataHandle;      { Simply return our stock text }
6 3106 --      DoSubstitution := FALSE;      { We didn't substitute }
6 3107 0- A  END;
6 3108 --
6 3109 --      {$S DlgRes}
6 3110 --      {-----+
6 3111 --      | Draw
6 3112 --      +-----}
6 3113 -- A  PROCEDURE TEditText.Draw (area: Rect); OVERRIDE;
6 3114 --
6 3115 --      VAR
6 3116 --          theRect:      Rect;
6 3117 --
6 3118 0- A  BEGIN
6 3119 --      IF fTextView <> NIL THEN
6 3120 1-      BEGIN
6 3121 --          fTextView.Draw(area);
6 3122 --          GetQDExtent(theRect);
6 3123 --          Adorn(theRect, fPenSize, fAdornment);
6 3124 1-      END
6 3125 --      ELSE
6 3126 --          INHERITED Draw(area);
6 3127 0- A  END;
6 3128 --
6 3129 --      {$IFC qDebug}
6 3130 --      {$S DlgFields}
6 3131 --      {-----+
6 3132 --      | Fields
6 3133 --      +-----}
6 3134 -- A  PROCEDURE TEditText.Fields (PROCEDURE DoToField (fieldName: Str255;
6 3135 --      fieldAddr: Ptr;
6 3136 --      fieldType: INTEGER)); OVERRIDE;
6 3137 0- A  BEGIN
6 3138 --      DoToField('TEditText', NIL, bClass);
6 3139 --      DoToField('fMaxChars', @fMaxChars, bInteger);
6 3140 --      DoToField('fControlChars', @fControlChars, bHexLongInt);
6 3141 --      DoToField('fTextView', @fTextView, bObject);
6 3142 --      INHERITED Fields(DoToField);
6 3143 0- A  END;
6 3144 --      {$ENDC}
6 3145 --
6 3146 --
6 3147 --      {$S DlgRes}
6 3148 --      {-----+
6 3149 --      | GetText
6 3150 --      +-----}
6 3151 -- A  PROCEDURE TEditText.GetText (VAR theText: Str255); OVERRIDE;
6 3152 --
6 3153 --      VAR
6 3154 --          theChars:      Handle;
6 3155 --          numberOfChars: INTEGER;
6 3156 --
6 3157 0- A  BEGIN
6 3158 --      IF fTextView = NIL THEN
6 3159 --          INHERITED GetText(theText)
6 3160 --      ELSE
6 3161 1-      BEGIN
6 3162 --          theChars := fTextView.ExtractText;
6 3163 --          numberOfChars := MIN(255, GetHandleSize(theChars));
6 3164 --      {$IFC qRangeCheck}{$R-}{$ENDC}
6 3165 --          theText[0] := CHR(numberOfChars);
6 3166 --      {$IFC qRangeCheck}{$R+}{$ENDC}
6 3167 --          BlockMove(Ptr(theChars^), Ptr(ORD4(@theText)+1), numberOfChars);

```

```

6 3168 --1      END;
6 3169 --0 A    END;
6 3170 --
6 3171 --
6 3172 --      {-----+
6 3173 --      | ImageText
6 3174 --      +-----}
6 3175 -- A    PROCEDURE TEditText.ImageText (text: Ptr; length: LONGINT; box: Rect; just: INTEGER); OVERRIDE;
6 3176 --
6 3177 --      CONST
6 3178 --          TEsysJust = $BAC;
6 3179 --
6 3180 --      VAR
6 3181 --          tmpHTE:      TEHandle;
6 3182 --
6 3183 --0 A    BEGIN
6 3184 --          IF length >= 0 THEN
6 3185 --1              BEGIN
6 3186 --                IF just = teJustLeft THEN
6 3187 --2                    BEGIN
6 3188 --                      IF gConfiguration.hasScriptManager THEN
6 3189 --                        just := GetSysJust
6 3190 --                      ELSE
6 3191 --                        ($IFC NOT qNeedsROM128K)
6 3192 --                          IF gConfiguration.hasROM128K THEN
6 3193 --                            ($ENDC)
6 3194 --                              just := PInteger(TEsysJust)^;
6 3195 --2                          END
6 3196 --                      ELSE IF just = teForceLeft THEN
6 3197 --                        just := teJustLeft;
6 3198 --                      { Could optimize for single lines, here... }
6 3199 --                      EraseRect(box);
6 3200 --                      tmpHTE := TENew(box, box);
6 3201 --                      tmpHTE^.just := just;
6 3202 --                      IF just = teJustLeft THEN
6 3203 --                        WITH tmpHTE^ DO
6 3204 --2                          BEGIN
6 3205 --                            crOnly := -1;
6 3206 --                            destRect.right := kMaxTEWidth;
6 3207 --2                          END;
6 3208 --                      TEsSetText(text, length, tmpHTE);
6 3209 --                      TEUpdate(box, tmpHTE);
6 3210 --                      TEDispose(tmpHTE);
6 3211 --1                      END;
6 3212 --0 A    END;
6 3213 --
6 3214 --      {$S DlgRes}
6 3215 --      {-----+
6 3216 --      | Locate
6 3217 --      +-----}
6 3218 -- A    PROCEDURE TEditText.Locate (h, v: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
6 3219 --
6 3220 --0 A    BEGIN
6 3221 --          INHERITED Locate(h, v, invalidate);
6 3222 --          IF fTextView <> NIL THEN
6 3223 --            fTextView.Locate(h, v, FALSE);
6 3224 --0 A    END;
6 3225 --
6 3226 --
6 3227 --      {$S DlgRes}
6 3228 --      {-----+
6 3229 --      | Resize
6 3230 --      +-----}
6 3231 -- A    PROCEDURE TEditText.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
6 3232 --
6 3233 --0 A    BEGIN
6 3234 --          INHERITED Resize(width, height, invalidate);
6 3235 --          IF fTextView <> NIL THEN
6 3236 --            fTextView.Resize(width, height, FALSE);
6 3237 --0 A    END;
6 3238 --
6 3239 --
6 3240 --      {$S DlgNonRes}
6 3241 --      {-----+
6 3242 --      | RestartEdit
6 3243 --      +-----}
6 3244 -- A    PROCEDURE TEditText.RestartEdit;
6 3245 --
6 3246 --0 A    BEGIN
6 3247 --          IF fTextView.Focus THEN
6 3248 --            TEsSetSelect(0, MAXINT, fTextView.fHTE);
6 3249 --0 A    END;
6 3250 --
6 3251 --
6 3252 --      {$S DlgNonRes}
6 3253 --      {-----+
6 3254 --      | SetJustification
6 3255 --      +-----}
6 3256 -- A    PROCEDURE TEditText.SetJustification (theJust: INTEGER; redraw: BOOLEAN);
6 3257 --0 A    BEGIN
6 3258 --          IF fTextView <> NIL THEN
6 3259 --            fTextView.SetJustification(theJust, kDontRedraw);
6 3260 --          INHERITED SetJustification(theJust, redraw);
6 3261 --0 A    END;
6 3262 --
6 3263 --      {$S DlgRes}
6 3264 --      {-----+
6 3265 --      | SetSelection
6 3266 --      +-----}

```

```

6 3267 -- A  PROCEDURE TEditText.SetSelection (selStart, selEnd: INTEGER; redraw: BOOLEAN);
6 3268 --
6 3269 0- A  BEGIN
6 3270 --      IF fTEView <> NIL THEN
6 3271 --          IF redraw & fTEView.Focus THEN
6 3272 --              TSetSelect(selStart, selEnd, fTEView.fHTE)
6 3273 --          ELSE
6 3274 --              SetSelect(selStart, selEnd, fTEView.fHTE);
6 3275 -0 A  END;
6 3276 --
6 3277 --      {$S DlgNonRes}
6 3278 --      {-----+
6 3279 --      |   SetText                               |
6 3280 --      +-----+}
6 3281 -- A  PROCEDURE TEditText.SetText (theText: Str255; redraw: BOOLEAN); OVERRIDE;
6 3282 --
6 3283 --      VAR
6 3284 --          currentText:   Str255;
6 3285 --          area:          Rect;
6 3286 --
6 3287 0- A  BEGIN
6 3288 --      IF fTEView <> NIL THEN
6 3289 1-          BEGIN
6 3290 --              GetText(currentText);
6 3291 --              IF currentText <> theText THEN
6 3292 2-                  BEGIN
6 3293 --                      fTEView.SetText(theText);
6 3294 --                      fTEView.RecalcText;
6 3295 --                      IF redraw & Focus THEN
6 3296 3-                          BEGIN
6 3297 --                              ControlArea(area);
6 3298 --                              EraseRect(area);
6 3299 --                              Draw(area);
6 3300 -3                              END;
6 3301 -2                          END;
6 3302 -1                      END
6 3303 --                      ELSE
6 3304 --                          INHERITED SetText(theText, redraw);
6 3305 -0 A  END;
6 3306 --
6 3307 --      {$S DlgRes}
6 3308 --      {-----+
6 3309 --      |   InstallSelection                       |
6 3310 --      +-----+}
6 3311 --
6 3312 -- A  PROCEDURE TEditText.InstallSelection (wasActive, beActive: BOOLEAN); OVERRIDE;
6 3313 --
6 3314 --      VAR
6 3315 --          itsDialogView:  TDialogView;
6 3316 --          myExtent:       VRect;
6 3317 --          minToSee:       Point;
6 3318 --          aString:        Str255;
6 3319 --
6 3320 0- A  BEGIN
6 3321 --      IF fTEView <> NIL THEN
6 3322 --          fTEView.InstallSelection(wasActive, beActive);
6 3323 -0 A  END;
6 3324 --
6 3325 --      {$S DlgNonRes}
6 3326 --      {-----+
6 3327 --      |   StartEdit                               |
6 3328 --      +-----+}
6 3329 --
6 3330 -- A  PROCEDURE TEditText.StartEdit (selectChars: BOOLEAN; theTEView: TDialogTEView);
6 3331 --
6 3332 --      VAR
6 3333 --          myExtent:       VRect;
6 3334 --          minToSee:       Point;
6 3335 --          itsWindow:      TWindow;
6 3336 --
6 3337 0- A  BEGIN
6 3338 --          theTEView.InstallEditText(SELF, selectChars);
6 3339 --
6 3340 --          itsWindow := GetWindow;           { Set the window's target, which sets }
6 3341 --          IF itsWindow <> NIL THEN           { _the application's target if it is }
6 3342 --              itsWindow.SetTarget(SELF);    { _the front window. }
6 3343 --
6 3344 --          GetExtent(myExtent);
6 3345 --          InsetVRect(myExtent, -10, -10);
6 3346 --          minToSee.h := MIN(fSize.h + 10, kMaxCoord);
6 3347 --          minToSee.v := MIN(fSize.v + 10, kMaxCoord);
6 3348 --
6 3349 --          RevealRect(myExtent, minToSee, kVisible);
6 3350 -0 A  END;
6 3351 --
6 3352 --      {$S DlgNonRes}
6 3353 --      {-----+
6 3354 --      |   StopEdit                               |
6 3355 --      +-----+}
6 3356 --
6 3357 -- A  FUNCTION TEditText.StopEdit: BOOLEAN;
6 3358 --
6 3359 --      VAR
6 3360 --          aString:        Str255;
6 3361 --
6 3362 0- A  BEGIN
6 3363 --          StopEdit := FALSE;
6 3364 --          IF Validate THEN
6 3365 1-              BEGIN

```

{ Assume the worst }

```

6 3366 --      GetText(aString);
6 3367 --      fTEView.InstallSelection(TRUE, FALSE);
6 3368 --      fTEView := NIL;
6 3369 --      SetText(aString, kDontRedraw);
6 3370 --      StopEdit := TRUE;
6 3371 --      END;
6 3372 -- A   END;
6 3373 --
6 3374 --
6 3375 --      {$S DlgRes}
6 3376 --      {-----+
6 3377 --      |   Validate   |
6 3378 --      +-----}
6 3379 -- A   FUNCTION TEditText.Validate: BOOLEAN;
6 3380 --
6 3381 -- A   BEGIN
6 3382 --      Validate := TRUE;
6 3383 -- A   END;
6 3384 --
6 3385 --
6 3386 --      (*****
6 3387 --      (*       T N u m b e r T e x t       *)
6 3388 --      (*****
6 3389 --
6 3390 --      {$IFC qProceduralViews}
6 3391 --      {$S DlgOpen}
6 3392 --      {-----+
6 3393 --      |   INumberText   |
6 3394 --      +-----}
6 3395 -- A   PROCEDURE TNumberText.INumberText (itsSuperView: TView; itsLocation, itsSize: VPoint;
6 3396 --      itsValue, itsMinimum, itsMaximum: LONGINT);
6 3397 --
6 3398 --      VAR
6 3399 --          aString:      Str255;
6 3400 -- A   BEGIN
6 3401 --      TEditText(itsSuperView, itsLocation, itsSize, 255);
6 3402 --      fMinimum := itsMinimum;
6 3403 --      fMaximum := itsMaximum;
6 3404 --      NumToString(itsValue, aString);
6 3405 --      SetText(aString, kDontRedraw);
6 3406 -- A   END;
6 3407 --      {$ENDC}
6 3408 --
6 3409 --
6 3410 --      {$IFC qTemplateViews}
6 3411 --      {$S DlgOpen}
6 3412 --      {-----+
6 3413 --      |   IRes   |
6 3414 --      +-----}
6 3415 -- A   PROCEDURE TNumberText.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
6 3416 --
6 3417 --      VAR
6 3418 --          aString:      Str255;
6 3419 --
6 3420 -- A   BEGIN
6 3421 --      INHERITED IRes(NIL, itsSuperView, itsParams);
6 3422 --
6 3423 --      WITH NumberTextTemplatePtr(itsParams)^ DO
6 3424 --      1-   BEGIN
6 3425 --          NumToString(value, aString);
6 3426 --          SetText(aString, kDontRedraw);
6 3427 --          fMinimum := minimum;
6 3428 --          fMaximum := maximum;
6 3429 --      1-   END;
6 3430 --
6 3431 --      OffsetPtr(itsParams, SIZEOF(NumberTextTemplate));
6 3432 -- A   END;
6 3433 --      {$ENDC}
6 3434 --
6 3435 --
6 3436 --      {$IFC qWriteTemplates}
6 3437 --      {$S MAWriteRes}
6 3438 --      {-----+
6 3439 --      |   WRes   |
6 3440 --      +-----}
6 3441 -- A   PROCEDURE TNumberText.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
6 3442 --
6 3443 --      VAR
6 3444 --          nmPtr:      NumberTextTemplatePtr;
6 3445 --
6 3446 -- A   BEGIN
6 3447 --      INHERITED WRes(theResource, itsParams);
6 3448 --
6 3449 --      nmPtr := NumberTextTemplatePtr(ExpandPtr(theResource, itsParams,
6 3450 --      SIZEOF(NumberTextTemplate)));
6 3451 --
6 3452 --      WITH nmPtr^ DO
6 3453 --      1-   BEGIN
6 3454 --          value := GetValue;
6 3455 --          minimum := fMinimum;
6 3456 --          maximum := fMaximum;
6 3457 --      1-   END;
6 3458 -- A   END;
6 3459 --
6 3460 --      {$S MAWriteRes}
6 3461 --      {-----+
6 3462 --      |   WriteRes   |
6 3463 --      +-----}
6 3464 -- A   PROCEDURE TNumberText.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;

```



```

6 3465 0- A BEGIN
6 3466 -- gWResSignature := 'nmbr'; gWResType := 'TNumberText';
6 3467 -- WRes(theResource, itsParams);
6 3468 -0 A END;
6 3469 -- {$ENDC}
6 3470 --
6 3471 --
6 3472 -- {$S DlgRes}
6 3473 -- {-----+
6 3474 -- | GetValue |
6 3475 -- +-----}
6 3476 - A FUNCTION TNumberText.GetValue: LONGINT;
6 3477 --
6 3478 -- VAR
6 3479 --     aString: Str255;
6 3480 --     theValue: LONGINT;
6 3481 --
6 3482 0- A BEGIN
6 3483 --     GetText(aString);
6 3484 --     StringToNum(aString, theValue);
6 3485 --     GetValue := theValue;
6 3486 -0 A END;
6 3487 --
6 3488 --
6 3489 -- {$S DlgNonRes}
6 3490 -- {-----+
6 3491 -- | SetValue |
6 3492 -- +-----}
6 3493 - A PROCEDURE TNumberText.SetValue (newValue: LONGINT; redraw: BOOLEAN);
6 3494 --
6 3495 -- VAR
6 3496 --     aString: Str255;
6 3497 --
6 3498 0- A BEGIN
6 3499 --     newValue := MAX(fMinimum, MIN(fMaximum, newValue));
6 3500 --     NumToString(newValue, aString);
6 3501 --     SetText(aString, redraw);
6 3502 -0 A END;
6 3503 --
6 3504 --
6 3505 -- {$S DlgNonRes}
6 3506 -- {-----+
6 3507 -- | Validate |
6 3508 -- +-----}
6 3509 - A FUNCTION TNumberText.Validate: BOOLEAN; OVERRIDE;
6 3510 --
6 3511 -- VAR
6 3512 --     theString: Str255;
6 3513 --     decRec: Decimal;
6 3514 --     extValue: Extended;
6 3515 --     index: INTEGER;
6 3516 --     validPrefix: BOOLEAN;
6 3517 --
6 3518 0- A BEGIN
6 3519 --     GetText(theString);
6 3520 --     IF LENGTH(theString) > 0 THEN
6 3521 1- BEGIN
6 3522 --         Validate := FALSE;
6 3523 --         index := 1;
6 3524 --         Str2Dec(theString, index, decRec, validPrefix);
6 3525 --         IF validPrefix AND (index > LENGTH(theString)) AND (decRec.exp >= 0) THEN
6 3526 2- BEGIN
6 3527 --             extValue := Dec2Num(decRec);
6 3528 --             Validate := (extValue >= fMinimum) & (extValue <= fMaximum);
6 3529 -2 END;
6 3530 -1 END
6 3531 --         ELSE
6 3532 --             Validate := TRUE;
6 3533 -0 A END;
6 3534 --
6 3535 -- {$IFC qDebug}
6 3536 -- {$S DlgFields}
6 3537 -- {-----+
6 3538 -- | Fields |
6 3539 -- +-----}
6 3540 - A PROCEDURE TNumberText.Fields (PROCEDURE DoToField (fieldName: Str255;
6 3541 --                                     fieldAddr: Ptr;
6 3542 --                                     fieldType: INTEGER)); OVERRIDE;
6 3543 0- A BEGIN
6 3544 --     DoToField('TNumberText', NIL, bClass);
6 3545 --     DoToField('fMinimum', @fMinimum, bLongInt);
6 3546 --     DoToField('fMaximum', @fMaximum, bLongInt);
6 3547 --     INHERITED Fields(DoToField);
6 3548 -0 A END;
6 3549 -- {$ENDC}
5 701 --
5 702 -- END.

```

```

7 1 --      {$P}
7 2 --      { UFailure.p }
7 3 --      { Copyright 1985-1988 by Apple Computer, Inc. All rights reserved. }
7 4 --
7 5 --      { Conditional compile flags used by this unit:
7 6 --
7 7 --          qDebug          True includes debugging code, false ignores debugging code.
7 8 --
7 9 --          qTrace          True surrounds trap patches with $D+ and $D++ (because
7 10 --                      the default is $D++), false skips these directives.
7 11 --
7 12 --      The settings of $D, $R and $N are imported from UMAUtil.
7 13 --      }
7 14 --
7 15 --
7 16 --      UNIT UFailure:
7 17 --
7 18 --      (*
7 19 --          T H E O R Y   O F   O P E R A T I O N
7 20 --
7 21 --      This unit implements the MacApp failure mechanism.
7 22 --
7 23 --      The failure mechanism is built around exception handlers. An exception
7 24 --      handler is a routine, generally local to some other routine, that is
7 25 --      called when a failure occurs and takes action to handle the failure.
7 26 --      An exception handler is of the form
7 27 --
7 28 --          PROCEDURE ExceptionHandler (error: OSErr; message: LONGINT);
7 29 --
7 30 --      where error is the error that caused the failure, and message identifies
7 31 --      the error message that may be displayed. Consider a routine that opens
7 32 --      a file, reads its contents, and closes the file. If a failure occurred
7 33 --      while reading the file, an exception handler would be needed to close the
7 34 --      file, as the rest of the routine will not be executed. (See the example
7 35 --      at the end of these comments.)
7 36 --
7 37 --      References to exception handlers are defined by the FailInfo record.
7 38 --      The exception handlers form a linked-list via the nextInfo field of
7 39 --      FailInfo. The linked list is a stack since new exception handlers
7 40 --      are added to the front of the list.
7 41 --
7 42 --      New exception handlers are added to the stack with the CatchFailures
7 43 --      procedure, and removed from the stack with the Success procedure. In
7 44 --      general you call CatchFailures to post an exception handler when an
7 45 --      error the application should handle might occur, and call Success to
7 46 --      remove the handler from the stack, after the handler is no longer
7 47 --      needed (i.e. the possibility of error no longer exists). Any failure
7 48 --      detected within the limits of the CatchFailures and subsequent Success
7 49 --      call results in the execution of the exception handler. (Failure does
7 50 --      not have to occur in the same routine as your call to CatchFailures.
7 51 --      The failure may occur in any routine called after CatchFailures but
7 52 --      before Success.)
7 53 --
7 54 --      When MacApp (or your code) determines that a failure has occurred, it
7 55 --      calls Failure. As a convenience, several procedures are provided
7 56 --      to check for standard kinds of failures and call Failure if needed.
7 57 --      These procedures are:
7 58 --
7 59 --          FailNIL           Calls Failure if its parameter is NIL.
7 60 --          FailOSErr        Calls Failure if its parameter is not noErr.
7 61 --          FailMemError     Calls Failure if MemError returns other than noErr.
7 62 --          FailResError     Calls Failure if ResError returns other than noErr.
7 63 --
7 64 --      When Failure is called, execution of the routine that called Failure is
7 65 --      terminated and the exception handler at the top of the stack is popped.
7 66 --      For each routine that was called after the handler was posted
7 67 --      to the stack, execution is terminated as though from an EXIT statement.
7 68 --      Then the exception handler is called. It generally cleans up for the
7 69 --      routine in which it is nested. Upon completion the next exception handler
7 70 --      is popped from the stack, repeating the process.
7 71 --
7 72 --      The error causing the failure, and a message code is passed to Failure.
7 73 --      For MacApp, the last exception handler on the stack is the one in
7 74 --      TApplication.PollEvent. It calls TApplication.ShowError, which calls
7 75 --      ErrorAlert, which decodes the message and displays an alert. You exception
7 76 --      handlers may set the message code to one more specific to your application
7 77 --      by calling FailNewMessage at the end of your exception handler.
7 78 --      FailNewMessage changes the message only if the current one is non-zero.
7 79 --      This has the effect of allowing those exception handlers closest to the
7 80 --      source of the error to set the message.
7 81 --
7 82 --      One last note about exception handlers: It is possible for an exception
7 83 --      handler to terminate exception processing by using a non-local GOTO to
7 84 --      jump back into the routine in which the exception handler is nested. This
7 85 --      is how MacApp keeps the application running when a failure occurs. The
7 86 --      last exception handler on the stack, in TApplication.PollEvent, uses a
7 87 --      GOTO to continue event processing.
7 88 --
7 89 --      The following is an example showing the use of exception handlers.
7 90 --      The ReadTheFile procedure reads the contents of a file, signalling
7 91 --      failure if the open, read, or close returns an error.
7 92 --
7 93 --          PROCEDURE ReadTheFile (fileName: Str255; volRefNum: INTEGER;
7 94 --                                VAR contents: ContentsRecord);
7 95 --      VAR
7 96 --          fi:          FailInfo;
7 97 --          count:       LONGINT;
7 98 --

```

```

7 99 --      PROCEDURE ExceptionHandler (error: OSErr; message: LONGINT);
7 100 --      VAR
7 101 --          err: OSErr;
7 102 --      BEGIN
7 103 --          err := FSClose(fileRefNum);      { Make sure the file is closed }
7 104 --          { Pass on my own error message... }
7 105 --          FailNewMessage(error, message, kMyErrorMessage);
7 106 --      END;
7 107 --
7 108 --      BEGIN
7 109 --          { If FSOpen fails we don't need to do any failure handling,
7 110 --            though the guy that called us probably will. }
7 111 --          FailOSErr(FSOpen(fileName, volRefNum, fileRefNum));
7 112 --
7 113 --          { Now that the file is open, if the read fails we must make
7 114 --            sure to close the file. }
7 115 --          CatchFailures(fi, ExceptionHandler);
7 116 --
7 117 --          count := SIZEOF(ContentsRecord);
7 118 --          FailOSErr(FSRead(fileRefNum, count, @contents));
7 119 --
7 120 --          { The file's been read so we don't need our exception handler
7 121 --            anymore. }
7 122 --          Success(fi);
7 123 --
7 124 --          { Again, the guy that called us will have to deal with this. }
7 125 --          FailOSErr(FSClose(fileRefNum));
7 126 --      END;
7 127 --
7 128 --      *)
7 129 --      INTERFACE
7 130 --
7 131 --      {-----+
7 132 --      | Uses
7 133 --      +-----+
7 134 --      USES
7 135 --          { $LOAD MacIntf.LOAD }
7 136 --          MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
7 137 --          { $LOAD }
7 138 --          UMAUtil
7 139 --          { $IFC qDebug }
7 140 --          , UWriteInWindow
7 141 --          , UTrace
7 142 --          { $ENDC }
7 143 --      ;
7 144 --
7 145 --      {-----+
7 146 --      | Global types
7 147 --      +-----+
7 148 --      TYPE
7 149 --          { used for the exception handling mechanism }
7 150 --          PFailInfo = ^FailInfo;
7 151 --          FailInfo = RECORD
7 152 --              regs:      ARRAY[1..11] OF LONGINT;
7 153 --              error:     INTEGER;
7 154 --              message:   LONGINT;
7 155 --              failA6:    LONGINT;
7 156 --              failPC:    LONGINT;
7 157 --              nextInfo:  PFailInfo;
7 158 --              { $IFC qDebug }
7 159 --              whoPC:     LONGINT;
7 160 --              { $ENDC }
7 161 --          END;
7 162 --
7 163 --      {-----+
7 164 --      | Global variables
7 165 --      +-----+
7 166 --      { $IFC qDebug }
7 167 --      VAR
7 168 --          gAskAboutAlloc:  BOOLEAN; { if TRUE ask about each allocation to force
7 169 --                                   failure }
7 170 --          gAskFailure:     BOOLEAN; { if TRUE ask about calls to
7 171 --                                   FailOSErr, FailResError, FailMemError,
7 172 --                                   and give user a chance to force error }
7 173 --      { $ENDC qDebug }
7 174 --
7 175 --
7 176 --      {-----+
7 177 --      | Global routines
7 178 --      +-----+
7 179 --      PROCEDURE InitUFailure;
7 180 --          { Initializes this unit. Must be called before using any features of
7 181 --            this unit. }
7 182 --
7 183 --      FUNCTION BuildMessage(lowWord, highWord: INTEGER): LONGINT;
7 184 --          INLINE $2E9F; { MOVE.L (A7)+, (A7) }
7 185 --          { Takes the 2 integers and combines them into a LONGINT. Note that the
7 186 --            low-order word is the first parameter. }
7 187 --
7 188 --      PROCEDURE CatchFailures(VAR fi: FailInfo;
7 189 --          PROCEDURE Handler(e: INTEGER; m: LONGINT));
7 190 --          { Call this to set up an exception handler. This pushes your handler onto
7 191 --            a stack of exception handlers. }
7 192 --
7 193 --      PROCEDURE Failure(error: INTEGER; message: LONGINT);
7 194 --          { Call this to signal a failure. Control will branch to the most recent
7 195 --            exception handler, which will be popped off the handler stack. }
7 196 --
7 197 --      PROCEDURE FailMemError;

```

```

7 198 --      { IF MemError <> noErr THEN Failure(MemError, 0); If you are using
7 199 --      assembler, then you should just test the return code from the Memory
7 200 --      Manager in DO by calling FailOSErr. (See the discussion of MemError in
7 201 --      Inside Macintosh.) }
7 202 --
7 203 --  PROCEDURE FailResError;
7 204 --      { IF ResError <> noErr THEN Failure(ResError, 0); (See Inside Macintosh.) }
7 205 --
7 206 --  PROCEDURE FailNewMessage(error: INTEGER; oldMessage, newMessage: LONGINT);
7 207 --      { This does:
7 208 --      IF oldMessage = 0 THEN
7 209 --          Failure(error, newMessage)
7 210 --      ELSE
7 211 --          Failure(error, oldMessage);
7 212 --      }
7 213 --
7 214 --  PROCEDURE FailNIL(p: UNIV Ptr);
7 215 --      { Call this with a pointer/handler; this signals Failure(memFullErr, 0) iff
7 216 --      the pointer is NIL. }
7 217 --
7 218 --  PROCEDURE FailOSErr(error: INTEGER);
7 219 --      { Call this with an OSErr; signals Failure(error, 0) iff error <> noErr. }
7 220 --
7 221 --  PROCEDURE SetInitHandler(handler: ProcPtr);
7 222 --      { Sets an initialization error handler; applications should not call
7 223 --      this, normally. }
7 224 --
7 225 --  PROCEDURE Success(VAR fi: FailInfo);
7 226 --      { Call this when you want to de-install your exception handler (pop 1
7 227 --      element off the handler stack). }
7 228 --
7 229 --  {$IFC qDebug}
7 230 --  PROCEDURE ProgramBreak(grievance: Str255);
7 231 --      { Causes a program break to the debug window, displaying grievance. }
7 232 --
7 233 --  PROCEDURE ProgramReport(grievance: Str255; break: BOOLEAN);
7 234 --      { Displays grievance in the debug window, and breaks if break is true. }
7 235 --  {$ENDC}
7 236 --
7 237 --
7 238 --  IMPLEMENTATION
7 239 --
7 240 --  {$I UFailure.incl.p}

```

```

8 1 --      {$P}
8 2 --      {UFailure.incl.p}
8 3 --      {Copyright 1985-1988 Apple Computer, Inc. All rights reserved.}
8 4 --
8 5 --      {-----+
8 6 --      | Local variables |
8 7 --      +-----}
8 8 --      VAR
8 9 --          {$Z+} {make gTopHandler accessible by assembly code}
8 10 --          gTopHandler: PFailInfo; {linked list of failure handlers}
8 11 --          {$Z-}
8 12 --          gInitHandler: ProcPtr;
8 13 --
8 14 --
8 15 --      {-----+
8 16 --      | External Declarations |
8 17 --      +-----}
8 18 -- A PROCEDURE CatchFailures (VAR fi: FailInfo;
8 19 --                          PROCEDURE Handler(e: INTEGER; m: LONGINT)); EXTERNAL;
8 20 --
8 21 -- A PROCEDURE DoFailure (pf: PFailInfo); EXTERNAL;
8 22 --
8 23 --      {-----+
8 24 --      | CallInitHandler |
8 25 --      +-----}
8 26 -- A PROCEDURE CallInitHandler (error: INTEGER; message: LONGINT; p: ProcPtr);
8 27 --      INLINE $205F, {MOVE.L (A7)+,A0 }
8 28 --      $4E90; {JMP (A0) }
8 29 --
8 30 --      {$S MAMain}
8 31 --      {-----+
8 32 --      | FailMemError |
8 33 --      +-----}
8 34 -- A PROCEDURE FailMemError;
8 35 --      VAR
8 36 --          e: OSErr;
8 37 --          {$IFC qDebug}
8 38 --          s: Str255;
8 39 --          {$ENDC}
8 40 -- A BEGIN
8 41 --      e := MemError;
8 42 --
8 43 --          {$IFC qDebug}
8 44 --          IF gAskFailure AND (e = noErr) AND CanReadLn THEN
8 45 --              BEGIN
8 46 --                  {$%+}
8 47 --                  GetMethodName(%_GetA6+4, s);
8 48 --                  {$%-}
8 49 --                  e := ReadInteger(CONCAT('FailMemError called by ', s, '. Enter return error: '));
8 50 --              END;
8 51 --          {$ENDC}
8 52 --
8 53 --          IF e <> noErr THEN
8 54 --              Failure(e, 0);
8 55 -- A END {FailMemError};
8 56 --
8 57 --
8 58 --      {$S MAMain}
8 59 --      {-----+
8 60 --      | FailNIL |
8 61 --      +-----}
8 62 -- A PROCEDURE FailNIL (p: UNIV Ptr);
8 63 -- A BEGIN
8 64 --      { no check for gAskFailure here, since we do this when objects are created. }
8 65 --      IF p = NIL THEN
8 66 --          Failure(memFullErr, 0);
8 67 -- A END {FailNIL};
8 68 --
8 69 --
8 70 --      {$S MAMain}
8 71 --      {-----+
8 72 --      | FailNewMessage |
8 73 --      +-----}
8 74 -- A PROCEDURE FailNewMessage (error: INTEGER; oldMessage, newMessage: LONGINT);
8 75 -- A BEGIN
8 76 --      IF oldMessage = 0 THEN
8 77 --          oldMessage := newMessage;
8 78 --          Failure(error, oldMessage);
8 79 -- A END {FailNewMessage};
8 80 --
8 81 --
8 82 --      {$S MAMain}
8 83 --      {-----+
8 84 --      | FailOSErr |
8 85 --      +-----}
8 86 -- A PROCEDURE FailOSErr (error: INTEGER);
8 87 --
8 88 --          {$IFC qDebug}
8 89 --          VAR
8 90 --              s: Str255;
8 91 --          {$ENDC}
8 92 --
8 93 -- A BEGIN
8 94 --          {$IFC qDebug}
8 95 --          IF gAskFailure AND (error = noErr) AND CanReadLn THEN
8 96 --              BEGIN
8 97 --                  {$%+}
8 98 --                  GetMethodName(%_GetA6+4, s);

```

```

8 99 --      {$%-}
8 100 --      error := ReadInteger(CONCAT('FailOSErr called by ', s, '. Enter return error: '));
8 101 -1      END;
8 102 --      {$ENDC}
8 103 --
8 104 --      IF error <> noErr THEN
8 105 --          Failure(error, 0);
8 106 -0 A  END (FailOSErr);
8 107 --
8 108 --
8 109 --      {$S MAMain}
8 110 --      {-----+
8 111 --      | FailResError
8 112 --      +-----}
8 113 -- A  PROCEDURE FailResError;
8 114 --      VAR
8 115 --          e: OSErr;
8 116 --      {$IFC qDebug}
8 117 --          s: Str255;
8 118 --      {$ENDC}
8 119 0- A  BEGIN
8 120 --          e := ResError;
8 121 --
8 122 --      {$IFC qDebug}
8 123 --          IF gAskFailure AND (e = noErr) AND CanReadLn THEN
8 124 1-      BEGIN
8 125 --          {$%+}
8 126 --              GetMethodName(%_GetA6+4, s);
8 127 --          {$%-}
8 128 --          e := ReadInteger(CONCAT('FailResError called by ', s, '. Enter return error: '));
8 129 -1      END;
8 130 --      {$ENDC}
8 131 --
8 132 --          IF e <> noErr THEN
8 133 --              Failure(e, 0);
8 134 -0 A  END (FailResError);
8 135 --
8 136 --
8 137 --      {$S MAMain}
8 138 --      {-----+
8 139 --      | Failure
8 140 --      +-----}
8 141 -- A  PROCEDURE Failure (error: INTEGER; message: LONGINT);
8 142 --      VAR
8 143 --          pf: PFailInfo;
8 144 --          ih: ProcPtr;
8 145 --          pc: LONGINT;
8 146 --      {$IFC qDebug}
8 147 --          cl: String8;
8 148 --          me: String8;
8 149 --          seg: INTEGER;
8 150 --          who: STRING[17];
8 151 --      {$ENDC}
8 152 0- A  BEGIN
8 153 --          pf := gTopHandler;
8 154 --
8 155 --          IF pf <> NIL THEN
8 156 1-      BEGIN
8 157 --      {$IFC qDebug}
8 158 --          pc := pf^.whoPC;
8 159 --          GetProcName(LONGINT(@pc), cl, me);
8 160 --          who := CONCAT(cl, '.', me);
8 161 --          IF cl = kSpace8 THEN
8 162 --              who[9] := ' ';
8 163 --
8 164 --          WriteLn('Failure caught by ', who);
8 165 --          WriteLn(' error = ', error:1, ' message = ', message:1,
8 166 --              ' (' , BSR(message, 16):1, '/', BAND(message, $0000FFFF):1, ')');
8 167 --      {$ENDC}
8 168 --
8 169 --          {pop the stack first, because calling the handler is likely to
8 170 --          result in a call to Failure}
8 171 --          gTopHandler := pf^.nextInfo;
8 172 --          pf^.error := error;
8 173 --          pf^.message := message;
8 174 --          DoFailure(pf);      {Go execute the failure handler}
8 175 -1      END
8 176 --      ELSE IF gInitHandler <> NIL THEN
8 177 1-      BEGIN
8 178 --          ih := gInitHandler;
8 179 --          gInitHandler := NIL;
8 180 --          CallInitHandler(error, message, ih);
8 181 --
8 182 --          ExitToShell;
8 183 -1      END
8 184 --      ELSE
8 185 1-      BEGIN
8 186 --      {$IFC qDebug}
8 187 --          ProgramBreak('Failure called, but no handler!');
8 188 --      {$ENDC}
8 189 -1      END;
8 190 -0 A  END (Failure);
8 191 --
8 192 --
8 193 --      {$S MAINit}
8 194 --      {$IFC qTrace}{$D+}{$ENDC}      {May be called before trace stuff is setup}
8 195 --      {-----+
8 196 --      | InitUFailure
8 197 --      +-----}

```

```

198 -- A PROCEDURE InitUFailure;
199 --
200 0- A BEGIN
201 --      gTopHandler := NIL;
202 --      gInitHandler := NIL;
203 --
204 --      {$IFC qDebug}
205 --      gAskAboutAlloc := FALSE;
206 --      gAskFailure := FALSE;
207 --      {$ENDC}
208 --
209 -0 A END {InitUFailure};
210 --      {$IFC qTrace}{$SD++}{$ENDC}
211 --
212 --
213 --      {$IFC qDebug}
214 --      {$IFC qTrace}{$SD+}{$ENDC}
215 --      {$S MADEbug}
216 --      {-----+
217 --      | ProgramBreak |
218 --      +-----}
219 -- A PROCEDURE ProgramBreak (grievance: Str255);
220 --      { ProgramBreak: Your app can call this when it comes to a situation that you do not expect
221 --      and cannot handle gracefully. It beeps and displays a message. If called before
222 --      there is a WriteLn window, it calls OBJFail, which goes into an infinite loop.
223 --      Otherwise, it enters our debugger. }
224 --      VAR
225 --          synthRec: RECORD
226 --              mode: INTEGER;
227 --              triplet: Tone;
228 --              endTriplet: Tone;
229 --              END;
230 --
231 0- A BEGIN
232 --      {$IFC FALSE}
233 --      WITH synthRec, triplet DO
234 1- BEGIN
235 --          mode := swMode;
236 --
237 --          count := 445;
238 --          amplitude := 100;
239 --          duration := 25;
240 --
241 --          endTriplet.count := 0;
242 --          endTriplet.amplitude := 0;
243 --          endTriplet.duration := 0;
244 -1 END;
245 --
246 --      StartSound(@synthRec, SIZEOF(synthRec), Pointer(-1));
247 --      {$ENDC}
248 --      SysBeep(2);
249 --
250 --      WWForceOutput(forceOn, forceUnchanged);
251 --      WriteLn('ProgramBreak: ', grievance);
252 --      WWEndForce;
253 --
254 --      {$IFC qTrace}
255 --      TRCBreak;
256 --      {$ELSEC}
257 --      OBJFail(kFailNone);
258 --      {$ENDC}
259 -0 A END {ProgramBreak};
260 --      {$IFC qTrace}{$SD++}{$ENDC}
261 --
262 --
263 --      {$IFC qTrace}{$SD+}{$ENDC}
264 --      {$S MADEbug}
265 --      {-----+
266 --      | ProgramReport |
267 --      +-----}
268 -- A PROCEDURE ProgramReport (grievance: Str255; break: BOOLEAN);
269 --
270 0- A BEGIN
271 --      WriteLn(grievance);
272 --      IF break THEN
273 --          TRCBreak;
274 -0 A END {ProgramReport};
275 --      {$IFC qTrace}{$SD++}{$ENDC}
276 --      {$ENDC qDebug}
277 --
278 --
279 --      {$S MAINit}
280 --      {$IFC qTrace}{$SD+}{$ENDC}
281 --      {-----+
282 --      | SetInitHandler |
283 --      +-----}
284 -- A PROCEDURE SetInitHandler (handler: ProcPtr);
285 0- A BEGIN
286 --      gInitHandler := handler;
287 -0 A END {SetInitHandler};
288 --      {$IFC qTrace}{$SD++}{$ENDC}
289 --
290 --
291 --      {We assume that the programmer passes in the correct FailInfo record: ie. the one that is the
292 --      top of the stack. If debugging is on, we check this fact.}
293 --      {$S MAMain}
294 --      {-----+
295 --      | Success |
296 --      +-----}

```

```
8 297 -- A  PROCEDURE Success (VAR fi: FailInfo);
8 298 0- A  BEGIN
8 299 --      { $IFC qDebug }
8 300 --      IF gTopHandler <> @fi THEN
8 301 1-          BEGIN
8 302 --              Write('gTopHandler = ');
8 303 --              WritePtr(gTopHandler);
8 304 --              Write('parameter = ');
8 305 --              WritePtr(@fi);
8 306 --              Writeln;
8 307 --              ProgramBreak('Problem with Success: too many or too few calls to Success');
8 308 -1          END;
8 309 --      { $ENDC }
8 310 --
8 311 --      gTopHandler := fi.nextInfo;
8 312 -0 A  END { Success };
8 313 --
7 241 --
7 242 --  END { UFailure }.
```



```

9 1 --      {$P}
9 2 --      {*****}
9 3 --      { *
9 4 --      { * File:          UGridView.p
9 5 --      { * Author:       Deb Orton
9 6 --      { * Date:         8 December 1987
9 7 --      { *
9 8 --      { * Description:   Provide an object for viewing data in a list or grid.
9 9 --      { *
9 10 --     { *
9 11 --     { * Copyright © 1987, 1988 by Apple Computer, Inc. All rights reserved.
9 12 --     {*****}
9 13 --
9 14 --
9 15 --     {*****}
9 16 --     { * HouseKeeping
9 17 --     {*****}
9 18 --
9 19 --     UNIT UGridView;
9 20 --
9 21 --     INTERFACE
9 22 --
9 23 --     USES
9 24 --         {$LOAD MacIntf.LOAD}
9 25 --         MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
9 26 --         {$LOAD UMacApp.LOAD}
9 27 --         UMAUtil, UViewCoords, UFailure, UMemory, UMenuSetup, UObject,
9 28 --         UList, UAssociation, UMacApp;
9 29 --
9 30 --
9 31 --     {*****}
9 32 --     { * Global Constants
9 33 --     {*****}
9 34 --
9 35 --     CONST
9 36 --
9 37 --         { * Template identifiers for views defined in this unit *}
9 38 --         kGridView      = 'TGridView';
9 39 --         kTextGridView   = 'TTextGridView';
9 40 --         kTextListView   = 'TTextListView';
9 41 --
9 42 --         { * For setting column/row width/height in only one row/col *}
9 43 --         kOneRow = 1;
9 44 --         kOneCol = 1;
9 45 --
9 46 --         { * Select Command Identifiers *}
9 47 --         cCellSelect    = 1;
9 48 --         cRowSelect      = 2;
9 49 --         cColumnSelect   = 3;
9 50 --         cVertexSelect   = 4;
9 51 --
9 52 --         { * Booleans for Adornment *}
9 53 --         kAdorn          = TRUE;
9 54 --         kDontAdorn       = FALSE;
9 55 --
9 56 --         { * Booleans for SetSelection *}
9 57 --         kExtend          = TRUE;
9 58 --         kDontExtend      = FALSE;
9 59 --
9 60 --         kHighlight       = TRUE;
9 61 --         kDontHighlight   = FALSE;
9 62 --
9 63 --         kSelect          = TRUE;
9 64 --         kDeSelect        = FALSE;
9 65 --
9 66 --         { * Booleans for CreateHighlightRgn *}
9 67 --         kWholeRect       = TRUE;
9 68 --         kNotWholeRect    = FALSE;
9 69 --
9 70 --
9 71 --     {*****}
9 72 --     { * Type and Object Definitions
9 73 --     {*****}
9 74 --
9 75 --     TYPE
9 76 --         GridCell      = Point; { * A cell is a QuickDraw point *}
9 77 --         GridViewPart   = (badChoice, inCell, inRow, inColumn, inVertex);
9 78 --         RunArrayChunk  = RECORD
9 79 --             count:    INTEGER;
9 80 --             size:     INTEGER;
9 81 --             END;
9 82 --
9 83 --         RunArray = ARRAY[0..100000] OF RunArrayChunk;
9 84 --         PRunArray = ^RunArray;
9 85 --         HRunArray = ^PRunArray;
9 86 --
9 87 --
9 88 --     {*****}
9 89 --     { * View Template Types
9 90 --     {*****}
9 91 --
9 92 --     GridViewTemplate = PACKED RECORD
9 93 --         numOfRows:    INTEGER;
9 94 --         numOfCols:    INTEGER;
9 95 --         rowHeight:    INTEGER;
9 96 --         colWidth:     INTEGER;
9 97 --         rowInset:     INTEGER;
9 98 --         colInset:     INTEGER;

```

```

9 99 --          adornRows:          BOOLEAN;
9 100 --          adornCols:          BOOLEAN;
9 101 --          singleSelection:    BOOLEAN;
9 102 --          filler:              0..8191;
9 103 --          END;
9 104 --      GridViewTemplatePtr = ^GridViewTemplate;
9 105 --
9 106 --
9 107 --      {*****}
9 108 --
9 109 --
9 110 --      TextGridViewTemplate = PACKED RECORD
9 111 --          itsFontFace:          Style;
9 112 --          itsFontSize:          INTEGER;
9 113 --          itsFontColor:         RGBColor;
9 114 --          itsFontName:          Str255; (* a variable length P-String *)
9 115 --      END;
9 116 --      TextGridViewTemplatePtr = ^TextGridViewTemplate;
9 117 --
9 118 --
9 119 --
9 120 --      {*****}
9 121 --      (* UGridView Objects *)
9 122 --      {*****}
9 123 --
9 124 --      {*****}
9 125 --      (* TGridView Object *)
9 126 --      {*****}
9 127 --      {*****}
9 128 --      {*****}
9 129 --
9 130 --      TGridView = OBJECT (TView)
9 131 --
9 132 --      {*****}
9 133 --      (* Instance Variables *)
9 134 --      {*****}
9 135 --
9 136 --          fMaxHorizontal:        LONGINT; (* horizontal pixels in view *)
9 137 --          fMaxVertical:          LONGINT; (* vertical pixels in view *)
9 138 --          fNumOfCols:            INTEGER; (* number of columns *)
9 139 --          fNumOfRows:            INTEGER; (* number of rows *)
9 140 --          fColWidths:            HRunArray; (* dynamic array containing col widths *)
9 141 --          fRowHeights:           HRunArray; (* dynamic array containing row heights *)
9 142 --          fAdornRows:            BOOLEAN; (* Draw adornment for rows? *)
9 143 --          fAdornCols:            BOOLEAN; (* Draw adornment for columns? *)
9 144 --          fRowInset:             INTEGER; (* Number of pixels between cell rows *)
9 145 --          fColInset:             INTEGER; (* Number of pixels between cell cols *)
9 146 --          fSelections:           RgnHandle; (* Region containing current selections *)
9 147 --          fWholeRect:            BOOLEAN; (* Is the new selection a rectangle? *)
9 148 --          fLastWholeRect:        BOOLEAN; (* Is the old selection a rectangle? *)
9 149 --          fSingleSelection:       BOOLEAN; (* only one cell selected at a time? *)
9 150 --          fAllowDimSelection:    BOOLEAN; (* allow the selection to be dimmed? *)
9 151 --          fHLRegion:             RgnHandle; (* Region of cells highlighted *)
9 152 --          fTmpSelRgn:            RgnHandle; (* Temporary region for selection *)
9 153 --          fTmpRgn:               RgnHandle; (* Temporary region for on-the-fly use *)
9 154 --          fTmpHLRgn:             RgnHandle; (* Temporary region for highlighting *)
9 155 --          fNumOfRowChunks:        INTEGER; (* # of Entries in the Row run array *)
9 156 --          fNumOfColChunks:        INTEGER; (* # of Entries in the Column run array *)
9 157 --          fLastRow:              INTEGER; (* These 4 fields used for run array *)
9 158 --          fLastRowChunkNum:       INTEGER; (* cacheing for rows... *)
9 159 --          fLastRowTotal:         LONGINT;
9 160 --          fLastRowIndex:         INTEGER;
9 161 --          fLastCol:              INTEGER; (* These 4 fields used for run array *)
9 162 --          fLastColChunkNum:       INTEGER; (* cacheing for columns... *)
9 163 --          fLastColTotal:         LONGINT;
9 164 --          fLastColIndex:         INTEGER;
9 165 --
9 166 --      {*****}
9 167 --      (* Initialization and Free Methods *)
9 168 --      {*****}
9 169 --
9 170 --      {IFC qProceduralViews}
9 171 --      PROCEDURE TGridView.IGridView (
9 172 --          itsDocument:          TDocument; (* Its document *)
9 173 --          itsSuperview:         TView; (* Its parent view *)
9 174 --          itsLocation:          VPoint; (* Top, Left in parent's coords *)
9 175 --          numofRows:            INTEGER; (* number of rows initially *)
9 176 --          numofCols:            INTEGER; (* number of columns initially *)
9 177 --          rowHeight:            INTEGER; (* Height for initial rows *)
9 178 --          colWidth:             INTEGER; (* width for initial columns *)
9 179 --          adornRows:            BOOLEAN; (* Adornment for Rows? *)
9 180 --          adornCols:            BOOLEAN; (* Adornment for Columns? *)
9 181 --          rowInset:             INTEGER; (* horizontal space between cells *)
9 182 --          colInset:             INTEGER; (* vertical space between cells *)
9 183 --          singleSelection:       BOOLEAN; (* single cell selection? *)
9 184 --
9 185 --          { Initialize the gridview. If numofRows or numofCols is non-zero
9 186 --            then the gridview is initialized to the specified size using
9 187 --            rowHeight and colWidth. If either adorn is kDontAdorn, then that
9 188 --            adorn will not be called. The "Insets" allow space between
9 189 --            cells and may be used to draw row and column adornments. }
9 190 --      {ENDC qProceduralViews}
9 191 --
9 192 --      {IFC qTemplateViews}
9 193 --      PROCEDURE TGridView.IRes (itsDocument: TDocument; itsSuperview: TView;
9 194 --          VAR itsParams: Ptr); OVERRIDE;
9 195 --          { Initialize the view template. }
9 196 --      {ENDC qTemplateViews}
9 197 --

```

```

9 198 --      {$IFC qWriteTemplates}
9 199 --          PROCEDURE TGridView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
9 200 --              { Write this object out as a view resource. }
9 201 --
9 202 --          PROCEDURE TGridView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
9 203 --              { Set up the type and signature of this object and call WRes. }
9 204 --      {$ENDC qWriteTemplates}
9 205 --
9 206 --          PROCEDURE TGridView.Free; OVERRIDE;
9 207 --              { Free the TGridView object }
9 208 --
9 209 --      {*****}
9 210 --      { * Methods to OVERRIDE * }
9 211 --      {*****}
9 212 --
9 213 --      {*****}
9 214 --      { * Inherited Methods (OVERRIDE) * }
9 215 --      {*****}
9 216 --
9 217 --          PROCEDURE TGridView.CalcMinSize (VAR minSize: VPoint); OVERRIDE;
9 218 --              { Sets the extent of the view. }
9 219 --
9 220 --          PROCEDURE TGridView.DoHighlightSelection (fromHL, toHL: HLState); OVERRIDE;
9 221 --              { Do highlighting }
9 222 --
9 223 --          FUNCTION TGridView.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
9 224 --              VAR hysteresis: Point) : TCommand; OVERRIDE;
9 225 --              { Handle mouse commands }
9 226 --
9 227 --          PROCEDURE TGridView.Draw (area: Rect); OVERRIDE;
9 228 --              { Calls DrawRangeOfCells and AdornRow/Col as appropriate. }
9 229 --
9 230 --
9 231 --      {*****}
9 232 --      { * UGridView Methods to OVERRIDE * }
9 233 --      {*****}
9 234 --
9 235 --          PROCEDURE TGridView.AdornCol (aCol: INTEGER; area: Rect);
9 236 --          PROCEDURE TGridView.AdornRow (aRow: INTEGER; area: Rect);
9 237 --              { If fAdornXxx is TRUE, these methods are called to draw the row and
9 238 --                column adornments. }
9 239 --
9 240 --          FUNCTION TGridView.CanSelectCell (aCell: GridCell) : BOOLEAN;
9 241 --              { This method checks to see if a cell can be selected. By default,
9 242 --                this method always returns TRUE. }
9 243 --
9 244 --          FUNCTION TGridView.CellExists (aCell: GridCell) : BOOLEAN;
9 245 --              { By default this method always returns TRUE. Cells that do not
9 246 --                exists will be skipped when drawing is done. }
9 247 --
9 248 --          PROCEDURE TGridView.DoHilite (hlRgn: RgnHandle; fromHL, toHL: HLState);
9 249 --              { This guys actually does the highlighting (inverts the rect).
9 250 --                Override it to get different highlighting behavior }
9 251 --
9 252 --          PROCEDURE TGridView.DrawRangeOfCells (startCell, stopCell: GridCell; aQDRect: Rect);
9 253 --              { This method calls DrawCell for each cell in need of re-drawing. }
9 254 --
9 255 --          PROCEDURE TGridView.DrawCell (aCell: GridCell; aQDRect: Rect);
9 256 --              { This method draws each individual cell. IT MUST BE OVERRIDDEN! }
9 257 --
9 258 --      {*****}
9 259 --      { * General UGridView Methods * }
9 260 --      {*****}
9 261 --
9 262 --          PROCEDURE TGridView.AllCellsDo (PROCEDURE DoToCell (aCell: GridCell));
9 263 --              { For every cell in the view perform "DoToCell". }
9 264 --
9 265 --          PROCEDURE TGridView.CellToVRect (aCell: GridCell; VAR aRect: VRect);
9 266 --              { Get the rectangle bounding a given cell. This includes the row
9 267 --                and column insets. }
9 268 --
9 269 --          PROCEDURE TGridView.ColToVRect (aCol: INTEGER; numCols: INTEGER; VAR aRect: VRect);
9 270 --          PROCEDURE TGridView.RowToVRect (aRow: INTEGER; numRows: INTEGER; VAR aRect: VRect);
9 271 --              { Get the rectangle bounding the given rows or columns. This includes
9 272 --                the cells in the row or column and the row or column insets. }
9 273 --
9 274 --          PROCEDURE TGridView.CreateHighlightRgn (oldRgn, newRgn: RgnHandle;
9 275 --              wholeRect: BOOLEAN);
9 276 --              { This guy goes through all the cells in the oldRgn and creates a
9 277 --                newRgn to be used for highlighting. }
9 278 --
9 279 --          PROCEDURE TGridView.DelColAt (aCol: INTEGER; numCols: INTEGER);
9 280 --          PROCEDURE TGridView.DelRowAt (aRow: INTEGER; numRows: INTEGER);
9 281 --              { Delete the specified columns. This causes a redraw of only the
9 282 --                necessary cells. }
9 283 --
9 284 --          PROCEDURE TGridView.DelColFirst (numCols: INTEGER);
9 285 --          PROCEDURE TGridView.DelRowFirst (numRows: INTEGER);
9 286 --              { Delete the specified columns. This causes a redraw of only the
9 287 --                necessary cells. }
9 288 --
9 289 --          PROCEDURE TGridView.DelColLast (numCols: INTEGER);
9 290 --          PROCEDURE TGridView.DelRowLast (numRows: INTEGER);
9 291 --              { Delete the specified columns. This causes a redraw of only the
9 292 --                necessary cells. }
9 293 --
9 294 --          PROCEDURE TGridView.EachCellDo (startCell, stopCell: GridCell;
9 295 --              PROCEDURE DoToCell (aCell: GridCell));
9 296 --              { For each cell in the rectangle defined by startCell and stopCell

```

```

9 297 --         perform "DoToCell". )
9 298 --
9 299 --     PROCEDURE TGridView.EachSelectedCellDo (PROCEDURE DoToCell (aCell: GridCell));
9 300 --         { For each cell in the selected region perform "DoToCell". }
9 301 --
9 302 --     PROCEDURE TGridView.EachInRgn (aRgn: RgnHandle;
9 303 --         PROCEDURE DoToCell (aCell: GridCell));
9 304 --         { For each cell in the specified region perform "DoToCell". }
9 305 --
9 306 --     FUNCTION TGridView.FindColChunk (aCol: INTEGER; VAR aChunkNum: INTEGER;
9 307 --         VAR theTotal: LONGINT; VAR indexInChunk: INTEGER): BOOLEAN;
9 308 --
9 309 --     FUNCTION TGridView.FindRowChunk (aRow: INTEGER; VAR aChunkNum: INTEGER;
9 310 --         VAR theTotal: LONGINT; VAR indexInChunk: INTEGER): BOOLEAN;
9 311 --
9 312 --     FUNCTION TGridView.FirstSelectedCell: GridCell;
9 313 --         { Returns the top left cell in the selection range, if any }
9 314 --
9 315 --     FUNCTION TGridView.GetColWidth (aCol: INTEGER) : INTEGER;
9 316 --     FUNCTION TGridView.GetRowHeight (aRow: INTEGER) : INTEGER;
9 317 --         { Return the width or height for the specified row or column. }
9 318 --
9 319 --     FUNCTION TGridView.IdentifyPoint (theQDPoint: Point;
9 320 --         VAR aRow, aCol: INTEGER): GridViewPart;
9 321 --         { Return the GridViewPart (inCell, inColumn, inRow, inVertex)
9 322 --         in which the specified point lies. The "vertex" is the area
9 323 --         where a row and column meet. }
9 324 --
9 325 --     PROCEDURE TGridView.InsColBefore (aCol: INTEGER; numCols: INTEGER; aWidth: INTEGER);
9 326 --     PROCEDURE TGridView.InsRowBefore (aRow: INTEGER; numRows: INTEGER; aHeight: INTEGER);
9 327 --         { Insert the specified rows or columns before the row/column given.
9 328 --         The heights/widths of the rows/columns are all set to the specified
9 329 --         Height/Width. }
9 330 --
9 331 --     PROCEDURE TGridView.InsColFirst (numCols: INTEGER; aWidth: INTEGER);
9 332 --     PROCEDURE TGridView.InsRowFirst (numRows: INTEGER; aHeight: INTEGER);
9 333 --         { Insert the specified rows or columns before the row/column given.
9 334 --         The heights/widths of the rows/columns are all set to the specified
9 335 --         Height/Width. }
9 336 --
9 337 --     PROCEDURE TGridView.InsColLast (numCols: INTEGER; aWidth: INTEGER);
9 338 --     PROCEDURE TGridView.InsRowLast (numRows: INTEGER; aHeight: INTEGER);
9 339 --         { Insert the specified rows or columns before the row/column given.
9 340 --         The heights/widths of the rows/columns are all set to the specified
9 341 --         Height/Width. }
9 342 --
9 343 --     PROCEDURE TGridView.InvalidateCell (aCell: GridCell);
9 344 --         { Cause a cell to be marked invalid (in need of re-drawing). }
9 345 --
9 346 --     PROCEDURE TGridView.InvalidateSelection;
9 347 --         { Cause the rectangle bounding the selections to be marked
9 348 --         invalid (in need of re-drawing). }
9 349 --
9 350 --     FUNCTION TGridView.IsCellSelected (aCell: GridCell) : BOOLEAN;
9 351 --         { Check if the specified cell is currently selected. }
9 352 --
9 353 --     FUNCTION TGridView.LastSelectedCell: GridCell;
9 354 --         { Returns the bottom right cell in the selection range, if any }
9 355 --
9 356 --     FUNCTION TGridView.SameCell (aCell, bCell: GridCell): BOOLEAN;
9 357 --         { Check if the cells are the same }
9 358 --
9 359 --     PROCEDURE TGridView.ScrollSelectionIntoView (redraw: BOOLEAN);
9 360 --         { Scroll the specified rectangle into view. }
9 361 --
9 362 --     PROCEDURE TGridView.SetColWidth (aCol: INTEGER; numCols: INTEGER; aWidth: INTEGER);
9 363 --     PROCEDURE TGridView.SetRowHeight (aRow: INTEGER; numRows: INTEGER; aHeight: INTEGER);
9 364 --         { Set the width/height of the specified rows/columns to that given. }
9 365 --
9 366 --     PROCEDURE TGridView.SelectCell (theCell: GridCell; extend, highlight, select: BOOLEAN);
9 367 --         { Set the current selection to the specified cell. This guy sets up a
9 368 --         region and then calls SetSelection. }
9 369 --
9 370 --     PROCEDURE TGridView.SetEmptySelection (highlight: BOOLEAN);
9 371 --         { Set the current selection to be empty or nothing. If highlight
9 372 --         is kHighlight then the appropriate highlighting is performed. }
9 373 --
9 374 --     PROCEDURE TGridView.SetSelection (theRegion: RgnHandle; extend, highlight, select: BOOLEAN);
9 375 --         { Set the current selections to the specified region. If extend is kExtend
9 376 --         then theRegion is "added" to that already selected, if highlight is
9 377 --         kHighlight then theRegion is highlighted as well. IF select is kSelect then
9 378 --         the region is selected, if kDeSelect then de-selected}
9 379 --
9 380 --     PROCEDURE TGridView.SetSelRect (theTop, theLeft, theBottom, theRight: INTEGER;
9 381 --         extend, highlight, select: BOOLEAN);
9 382 --         { Set the current selections to the specified rectangle. and
9 383 --         call SetSelection. }
9 384 --
9 385 --     PROCEDURE TGridView.SetSingleSelection (theSetting: BOOLEAN);
9 386 --         { If TRUE then only one item/cell can be selected at a time }
9 387 --
9 388 --     FUNCTION TGridView.VPointToCell (aPoint: VPoint): GridCell;
9 389 --         { Determine the cell in which a given point lies. }
9 390 --
9 391 --     {*****}
9 392 --     { * Debugging Methods * }
9 393 --     {*****}
9 394 --
9 395 --     {IFC qDebug}

```

```

9 396 --      PROCEDURE TGridView.Fields (
9 397 --          PROCEDURE DoToField (
9 398 --              fieldName: Str255;
9 399 --              fieldAddr: Ptr;
9 400 --              fieldType: INTEGER); OVERRIDE;
9 401 --      ($ENDC)
9 402 --
9 403 --      END; (* TGridView *)
9 404 --
9 405 --
9 406 --      {*****}
9 407 --      {*****}
9 408 --      (* TTextGridView Object Sub-Class *)
9 409 --      {*****}
9 410 --      {*****}
9 411 --
9 412 --      TTextGridView = OBJECT (TGridView)
9 413 --
9 414 --      {*****}
9 415 --      (* Instance Variables *)
9 416 --      {*****}
9 417 --
9 418 --      fTextStyle:    TextStyle; (* The text style (color, size, etc. ) *)
9 419 --      fLineHeight:   INTEGER; (* height of each item including leading *)
9 420 --      fLineAscent:   INTEGER; (* pos. of baseline relative to top of line *)
9 421 --
9 422 --      {*****}
9 423 --      (* Initialization and Free Methods *)
9 424 --      {*****}
9 425 --
9 426 --      ($IFC qProceduralViews)
9 427 --          PROCEDURE TTextGridView.ITextGridView (
9 428 --              itsDocument:    TDocument; (* Its document *)
9 429 --              itsSuperview:   TView; (* Its parent view *)
9 430 --              itsLocation:    VPoint; (* Top, Left in parent's coords *)
9 431 --              numofRows:      INTEGER; (* Number of rows initially *)
9 432 --              numofCols:      INTEGER; (* Number of columns initially *)
9 433 --              colWidth:       INTEGER; (* Width of items in the columns *)
9 434 --              adornRows:      BOOLEAN; (* Adornment for Rows? *)
9 435 --              adornCols:      BOOLEAN; (* Adornment for Columns? *)
9 436 --              rowInset:       INTEGER; (* horizontal space between cells *)
9 437 --              colInset:       INTEGER; (* vertical space between cells *)
9 438 --              singleSelection: BOOLEAN; (* single cell selection? *)
9 439 --              itsTextStyle:   TextStyle; (* size, color, etc. font info *)
9 440 --
9 441 --              { Initialize the TextGridView. If numofRows or numofCols is non-zero
9 442 --                  then the gridview is initialized to the specified size. The row
9 443 --                  and height of the cells is determine by the font size, style etc.
9 444 --                  The default size may be changed with calls to the the SetRowHeight
9 445 --                  and SetColWidth methods. If either adorn is kDontAdorn, then that
9 446 --                  adorn will not be called. The "Insets" allow space between
9 447 --                  cells and may be used to draw row and column adornments. }
9 448 --          ($ENDC qProceduralViews)
9 449 --
9 450 --      ($IFC qTemplateViews)
9 451 --          PROCEDURE TTextGridView.IRes (itsDocument: TDocument; itsSuperview: TView;
9 452 --              VAR itsParams: Ptr); OVERRIDE;
9 453 --              { Initialize the view template. }
9 454 --          ($ENDC qTemplateViews)
9 455 --
9 456 --      ($IFC qWriteTemplates)
9 457 --          PROCEDURE TTextGridView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
9 458 --              { Write this object out as a view resource. }
9 459 --
9 460 --          PROCEDURE TTextGridView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
9 461 --              { Set up the type and signature of this object and call WRes. }
9 462 --          ($ENDC qWriteTemplates)
9 463 --
9 464 --
9 465 --      {*****}
9 466 --      (* Methods To OVERRIDE *)
9 467 --      {*****}
9 468 --
9 469 --      PROCEDURE TTextGridView.GetText (aCell: GridCell; VAR aString: Str255);
9 470 --          { This routine gets the text to be draw in a given cell. IT MUST
9 471 --              BE OVERRIDDEN! }
9 472 --
9 473 --
9 474 --      {*****}
9 475 --      (* General Methods *)
9 476 --      {*****}
9 477 --
9 478 --
9 479 --      PROCEDURE TTextGridView.DrawCell (aCell: GridCell; aQDRect: Rect); OVERRIDE;
9 480 --          { Calls GetText and then draws the text }
9 481 --
9 482 --      FUNCTION TTextGridView.Focus : BOOLEAN; OVERRIDE;
9 483 --          { Calls inherited focus and calls SetUpFont }
9 484 --
9 485 --      PROCEDURE TTextGridView.SetUpFont;
9 486 --          {Sets up the font for this view. }
9 487 --
9 488 --      PROCEDURE TTextGridView.SetPen;
9 489 --          {Sets the font characteristics of the current port. Called at
9 490 --              the beginning of the Draw method.}
9 491 --
9 492 --      ($IFC qDebug)
9 493 --          PROCEDURE TTextGridView.Fields (
9 494 --              PROCEDURE DoToField (

```

```

9 495 --      fieldName: Str255;
9 496 --      fieldAddr: Ptr;
9 497 --      fieldType: INTEGER); OVERRIDE;
9 498 --  {$ENDC}
9 499 --
9 500 --      END; (* TTextGridView *)
9 501 --
9 502 --
9 503 --  {*****}
9 504 --  {*****}
9 505 --  (* TTextListView Object Sub-Class *)
9 506 --  {*****}
9 507 --  {*****}
9 508 --
9 509 --      TTextListView = OBJECT (TTextGridView)
9 510 --
9 511 --      {*****}
9 512 --      (* Instance Variables *)
9 513 --      {*****}
9 514 --
9 515 --      (* none *)
9 516 --
9 517 --      {*****}
9 518 --      (* Initialization and Free Methods *)
9 519 --      {*****}
9 520 --
9 521 --  {$IFC qProceduralViews}
9 522 --      PROCEDURE TTextListView.ITextListView (
9 523 --          itsDocument: TDocument;      (* Its document *)
9 524 --          itsSuperview: TView;          (* Its parent view *)
9 525 --          itsLocation: VPoint;          (* Top, Left in parent's coords *)
9 526 --          numOfItems: INTEGER;          (* Number of items initially *)
9 527 --          adornRows: BOOLEAN;           (* Draw the row adornments? *)
9 528 --          adornCols: BOOLEAN;           (* Draw the col adornment? *)
9 529 --          rowInset: INTEGER;            (* Amount to inset the rows *)
9 530 --          colInset: INTEGER;            (* Amount to inset the column *)
9 531 --          singleSelection: BOOLEAN;      (* single cell selection? *)
9 532 --          itsTextStyle: TextStyle;      (* size, color, etc. font info *)
9 533 --
9 534 --          { Initialize the TextListView. If numOfItems is non-zero
9 535 --            then the listview is initialized to the specified size. The width
9 536 --            and height of the items are determined by the font size, style etc.
9 537 --            The default size may be changed with calls to the the SetItemHeight
9 538 --            and SetItemWidth methods. The "Insets" allow space between items. }
9 539 --      {$ENDC qProceduralViews}
9 540 --
9 541 --  {$IFC qWriteTemplates}
9 542 --      PROCEDURE TTextListView.WriteRes (theResource: ViewRsrcHndl;
9 543 --          VAR itsParams: Ptr); OVERRIDE;
9 544 --      { Set up the type and signature of this object and call WRes. }
9 545 --  {$ENDC qWriteTemplates}
9 546 --
9 547 --
9 548 --  {*****}
9 549 --  (* Methods To OVERRIDE *)
9 550 --  {*****}
9 551 --
9 552 --      PROCEDURE TTextListView.GetItemText (anItem: INTEGER; VAR aString: Str255);
9 553 --      { Get the text to be drawn for a given item. THIS METHOD MUST BE
9 554 --        OVERRIDDEN !! }
9 555 --
9 556 --      FUNCTION TTextListView.CanSelectItem (anItem: INTEGER) : BOOLEAN;
9 557 --      { Determine if the item is select-able }
9 558 --
9 559 --  {*****}
9 560 --  (* General Methods *)
9 561 --  {*****}
9 562 --      PROCEDURE TTextListView.AllItemsDo (PROCEDURE DoToItem (anItem: INTEGER));
9 563 --      { For all the items in the view perform "DoToItem". }
9 564 --
9 565 --      FUNCTION TTextListView.CanSelectCell (aCell: GridCell) : BOOLEAN; OVERRIDE;
9 566 --      { This method checks to see if a cell can be selected. This method
9 567 --        simply calls CanSelectItem }
9 568 --
9 569 --      PROCEDURE TTextListView.DeleteItemAt (anItem: INTEGER; numOfItems: INTEGER);
9 570 --      PROCEDURE TTextListView.DeleteItemFirst (numOfItems: INTEGER);
9 571 --      PROCEDURE TTextListView.DeleteItemLast (numOfItems: INTEGER);
9 572 --      { Delete the specified items. This causes a redraw of only the
9 573 --        necessary cells. }
9 574 --
9 575 --      PROCEDURE TTextListView.EachItemDo (start, stop: INTEGER;
9 576 --          PROCEDURE DoToItem (anItem: INTEGER));
9 577 --      { For each item in the specified range perform "DoToItem". }
9 578 --
9 579 --      PROCEDURE TTextListView.EachSelectedItemDo (PROCEDURE DoToItem (anItem: INTEGER));
9 580 --      { For each item selected perform "DoToItem". }
9 581 --
9 582 --      FUNCTION TTextListView.GetItemHeight (anItem: INTEGER) : INTEGER;
9 583 --      { Return the height of the specified item. }
9 584 --
9 585 --      FUNCTION TTextListView.GetItemWidth: INTEGER;
9 586 --      { Return the width of the view. }
9 587 --
9 588 --      PROCEDURE TTextListView.GetText (aCell: GridCell; VAR aString: Str255); OVERRIDE;
9 589 --      { Calls GetItemText with an item number. }
9 590 --
9 591 --      PROCEDURE TTextListView.InsertItemBefore (anItem: INTEGER; numOfItems: INTEGER);
9 592 --      PROCEDURE TTextListView.InsertItemFirst (numOfItems: INTEGER);
9 593 --      PROCEDURE TTextListView.InsertItemLast (numOfItems: INTEGER);

```

```

9 594 --      ( Insert the specified items. This causes a redraw of only the
9 595 --      necessary cells. )
9 596 --
9 597 --      FUNCTION TTextView.IsItemSelected (anItem: INTEGER) : BOOLEAN;
9 598 --      { Check if the specified item is currently selected. }
9 599 --
9 600 --      PROCEDURE TTextView.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
9 601 --      { Resize the view. }
9 602 --
9 603 --      PROCEDURE TTextView.SelectCell (theCell: GridCell; extend, highlight, select: BOOLEAN); OVERRIDE;
9 604 --      { Calls SelectItem with the appropriate item number }
9 605 --
9 606 --      PROCEDURE TTextView.SelectItem (anItem: INTEGER; extend, highlight, select: BOOLEAN);
9 607 --      { Select the given item. If extend is true then the item is added to the
9 608 --      current selection. Otherwise the current selection is deselected.
9 609 --      If highlight is true then the selected item is highlighted.}
9 610 --
9 611 --      PROCEDURE TTextView.SetItemHeight (anItem: INTEGER; numItems: INTEGER;
9 612 --      aHeight: INTEGER);
9 613 --      { Set the height of an item to be different from the default. This
9 614 --      changes the height of only the specified items. }
9 615 --
9 616 --      PROCEDURE TTextView.SetItemWidth (aWidth: INTEGER);
9 617 --      { Set the width of the view to aWidth. (This changes the width of all
9 618 --      the items. )
9 619 --
9 620 --      ($IFC qDebug)
9 621 --      PROCEDURE TTextView.Fields (
9 622 --      PROCEDURE DoToField (
9 623 --      fieldName: Str255;
9 624 --      fieldAddr: Ptr;
9 625 --      fieldType: INTEGER)); OVERRIDE;
9 626 --      ($ENDC)
9 627 --
9 628 --      END;      (* TTextView *)
9 629 --
9 630 --
9 631 --
9 632 --      {*****}
9 633 --      { * Support Objects * }
9 634 --      {*****}
9 635 --
9 636 --      {*****}
9 637 --      { * TGridSelectCommand * }
9 638 --      {*****}
9 639 --      TGridSelectCommand = OBJECT (TCommand)
9 640 --      fGridView:      TGridView;      (* The associated GridView *)
9 641 --      fShiftKey:      BOOLEAN;      (* Shift Key down? *)
9 642 --      fCmdKey:        BOOLEAN;      (* Command Key down? *)
9 643 --      fOldRgn:        RgnHandle;      (* region of old selected cells *)
9 644 --      fNewRgn:        RgnHandle;      (* region of new selected cells *)
9 645 --
9 646 --      PROCEDURE TGridSelectCommand.IGridSelectCommand (itsCmdNumber: CmdNumber;
9 647 --      itsView: TGridView; theShiftKey, theCmdKey: BOOLEAN);
9 648 --      { Initialize the command object. }
9 649 --
9 650 --      PROCEDURE TGridSelectCommand.Free; OVERRIDE;
9 651 --      { Free the command object. }
9 652 --
9 653 --      PROCEDURE TGridSelectCommand.TrackFeedback (anchorPoint, nextPoint: VPoint;
9 654 --      turnItOn, mouseDidMove: BOOLEAN); OVERRIDE;
9 655 --      { Track the mouse while the button is down. }
9 656 --
9 657 --      ($IFC qDebug)
9 658 --      PROCEDURE TGridSelectCommand.Fields (
9 659 --      PROCEDURE DoToField (
9 660 --      fieldName: Str255;
9 661 --      fieldAddr: Ptr;
9 662 --      fieldType: INTEGER)); OVERRIDE;
9 663 --      ($ENDC)
9 664 --
9 665 --      END;      (* TGridSelectCommand *)
9 666 --
9 667 --
9 668 --      {*****}
9 669 --      { * TCellSelectCommand * }
9 670 --      {*****}
9 671 --
9 672 --      TCellSelectCommand = OBJECT (TGridSelectCommand)
9 673 --      fOldCell:      GridCell;      (* The last cell selected *)
9 674 --      fNewMouseDown: BOOLEAN;      (* determine if this is a new mouse down *)
9 675 --      fKeepSelecting: BOOLEAN;      (* select all cells? (Cmd Key) *)
9 676 --
9 677 --      PROCEDURE TCellSelectCommand.ICellSelectCommand (itsView: TGridView;
9 678 --      aCell: GridCell; theShiftKey, theCmdKey: BOOLEAN);
9 679 --      { Initialize the command object. }
9 680 --
9 681 --      PROCEDURE TCellSelectCommand.HandleShiftKeyDown (aView: TGridView;
9 682 --      aCell, anchorCell: GridCell; mouseDidMove: BOOLEAN);
9 683 --      { Provides support for TrackFeedback and TrackMouse when the shift
9 684 --      key is down with the mouse button. }
9 685 --
9 686 --      PROCEDURE TCellSelectCommand.TrackFeedback (anchorPoint, nextPoint: VPoint;
9 687 --      turnItOn, mouseDidMove: BOOLEAN); OVERRIDE;
9 688 --      { Track the mouse while the button is down. }
9 689 --
9 690 --      FUNCTION TCellSelectCommand.TrackMouse (aTrackPhase: TrackPhase;
9 691 --      VAR anchorPoint, previousPoint, nextPoint: VPoint;
9 692 --      mouseDidMove: BOOLEAN): TCommand; OVERRIDE;

```

```

9 693 --      { Track the mouse when the button is up. }
9 694 --
9 695 --      { $IFC qDebug }
9 696 --      PROCEDURE TCellSelectCommand.Fields (
9 697 --          PROCEDURE DoToField (
9 698 --              fieldName: Str255;
9 699 --              fieldAddr: Ptr;
9 700 --              fieldType: INTEGER)); OVERRIDE;
9 701 --      { $ENDC }
9 702 --
9 703 --
9 704 --      END;      { * TCellSelectCommand * }
9 705 --
9 706 --
9 707 --      {*****}
9 708 --      { * TRowSelectCommand * }
9 709 --      {*****}
9 710 --
9 711 --      TRowSelectCommand = OBJECT (TGridSelectCommand)
9 712 --          fTheRow:      INTEGER;      { * The row selected * }
9 713 --
9 714 --          PROCEDURE TRowSelectCommand.IRowSelectCommand (itsView: TGridView;
9 715 --              aRow: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
9 716 --              { Initialize the command object. }
9 717 --
9 718 --      { $IFC qDebug }
9 719 --      PROCEDURE TRowSelectCommand.Fields (
9 720 --          PROCEDURE DoToField (
9 721 --              fieldName: Str255;
9 722 --              fieldAddr: Ptr;
9 723 --              fieldType: INTEGER)); OVERRIDE;
9 724 --      { $ENDC }
9 725 --
9 726 --
9 727 --      END;      { * TRowSelectCommand * }
9 728 --
9 729 --
9 730 --      {*****}
9 731 --      { * TColSelectCommand * }
9 732 --      {*****}
9 733 --
9 734 --      TColSelectCommand = OBJECT (TGridSelectCommand)
9 735 --          fTheColumn:    INTEGER;      { * The column selected * }
9 736 --
9 737 --          PROCEDURE TColSelectCommand.IColSelectCommand (itsView: TGridView;
9 738 --              aColumn: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
9 739 --              { Initialize the command object. }
9 740 --
9 741 --      { $IFC qDebug }
9 742 --      PROCEDURE TColSelectCommand.Fields (
9 743 --          PROCEDURE DoToField (
9 744 --              fieldName: Str255;
9 745 --              fieldAddr: Ptr;
9 746 --              fieldType: INTEGER)); OVERRIDE;
9 747 --      { $ENDC }
9 748 --
9 749 --      END; { * TColSelectCommand * }
9 750 --
9 751 --
9 752 --      {*****}
9 753 --      { * TVertexSelectCommand * }
9 754 --      {*****}
9 755 --
9 756 --      TVertexSelectCommand = OBJECT (TGridSelectCommand)
9 757 --          fTheColumn:    INTEGER;      { * The column selected * }
9 758 --          fTheRow:      INTEGER;      { * The column selected * }
9 759 --
9 760 --          PROCEDURE TVertexSelectCommand.IVertexSelectCommand (itsView: TGridView;
9 761 --              aRow, aColumn: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
9 762 --              { Initialize the command object. }
9 763 --
9 764 --      { $IFC qDebug }
9 765 --      PROCEDURE TVertexSelectCommand.Fields (
9 766 --          PROCEDURE DoToField (
9 767 --              fieldName: Str255;
9 768 --              fieldAddr: Ptr;
9 769 --              fieldType: INTEGER)); OVERRIDE;
9 770 --      { $ENDC }
9 771 --
9 772 --      END; { * TVertexSelectCommand * }
9 773 --
9 774 --
9 775 --      {*****}
9 776 --      { * Implementation * }
9 777 --      {*****}
9 778 --
9 779 --      IMPLEMENTATION
9 780 --
9 781 --      { $I UGridView.incl.p }

```



```

10 1 --      {$P}
10 2 --      {*****}
10 3 --      {*
10 4 --      {* File:          UGridView.incl.p
10 5 --      {* Author:       Deb Orton
10 6 --      {* Date:        8 December 1987
10 7 --      {*
10 8 --      {*
10 9 --      {* Copyright © 1987, 1988 by Apple Computer, Inc. All rights reserved.
10 10 --     {*****}
10 11 --
10 12 --
10 13 --     {*****}
10 14 --     {*****}
10 15 --     {* TGridView Object
10 16 --     {*****}
10 17 --     {*****}
10 18 --
10 19 --     {*****}
10 20 --     {* TGridView.IGridView
10 21 --     {* Initialize the gridview. If numOfRows or numOfCols is non-zero
10 22 --     {* then the gridview is initialized to the specified size using
10 23 --     {* RowHeight and ColWidth. If either adorn is false, then that
10 24 --     {* adorn method will not be called. The "Insets" allow space between
10 25 --     {* cells and may be used to draw row and column adornments.
10 26 --     {*****}
10 27 --     {$IFC qProceduralViews}
10 28 --     {$S GVOpen}
10 29 -- A  PROCEDURE TGridView.IGridView (
10 30 --         itsDocument:    TDocument; (* Its document
10 31 --         itsSuperView:    TView;     (* Its parent view
10 32 --         itsLocation:     VPoint;    (* Top, Left in parent's coords
10 33 --         numOfRows:       INTEGER;   (* Number of rows initially
10 34 --         numOfCols:       INTEGER;   (* Number of columns initially
10 35 --         rowHeight:       INTEGER;   (* Height of initial rows
10 36 --         colWidth:        INTEGER;   (* Height of initial columns
10 37 --         adornRows:       BOOLEAN;   (* Adornment for Rows?
10 38 --         adornCols:       BOOLEAN;   (* Adornment for Columns?
10 39 --         rowInset:        INTEGER;   (* horizontal space between cells
10 40 --         colInset:        INTEGER;   (* vertical space between cells
10 41 --         singleSelection: BOOLEAN;   (* single cell selection?
10 42 --
10 43 --     VAR
10 44 --         itsSize:         VPoint;
10 45 --         aHRunArray:      HRunArray;
10 46 --         tmpRgn:          RgnHandle;
10 47 --
10 48 -- A  BEGIN
10 49 --
10 50 --         fNumOfRows := 0;
10 51 --         fNumOfCols := 0;
10 52 --
10 53 --         fAdornRows := adornRows;
10 54 --         fAdornCols := adornCols;
10 55 --
10 56 --         (* Make sure the insets are evenly divided between top/bottom or left/right *)
10 57 --         IF ODD (rowInset) THEN
10 58 --             fRowInset := rowInset + 1
10 59 --         ELSE
10 60 --             fRowInset := rowInset;
10 61 --
10 62 --         IF ODD (colInset) THEN
10 63 --             fColInset := colInset + 1
10 64 --         ELSE
10 65 --             fColInset := colInset;
10 66 --
10 67 --
10 68 --         (* "Munger" is used to grow and shrink the array containing row heights and
10 69 --         * column widths. The fRowHeights and fColWidths are initially handles
10 70 --         * to empty arrays.
10 71 --         *)
10 72 --
10 73 --         aHRunArray := HRunArray (NewPermHandle (0));
10 74 --         FailNil (aHRunArray);
10 75 --         fColWidths := aHRunArray;
10 76 --
10 77 --         aHRunArray := HRunArray (NewPermHandle (0));
10 78 --         FailNil (aHRunArray);
10 79 --         fRowHeights := aHRunArray;
10 80 --
10 81 --
10 82 --         (* for cacheing run array values *)
10 83 --         fLastRow := 0;
10 84 --         fLastRowChunkNum := 0;
10 85 --         fLastRowTotal := 0;
10 86 --         fLastRowIndex := 1;
10 87 --         fNumOfRowChunks := 0;
10 88 --
10 89 --         fLastCol := 0;
10 90 --         fLastColChunkNum := 0;
10 91 --         fLastColTotal := 0;
10 92 --         fLastColIndex := 1;
10 93 --         fNumOfColChunks := 0;
10 94 --
10 95 --         fMaxHorizontal := 0;
10 96 --         fMaxVertical := 0;
10 97 --
10 98 --         itsSize.v := 0;

```

```

10 99 --      itsSize.h := 0;
10 100 --
10 101 --      IView (itsDocument, itsSuperview, itsLocation, itsSize,
10 102 --           sizeVariable, sizeVariable);
10 103 --
10 104 --      tmpRgn := MakeNewRgn;
10 105 --      fSelections := tmpRgn;      (* region to hold current selections *)
10 106 --
10 107 --      tmpRgn := MakeNewRgn;
10 108 --      fHLRegion := tmpRgn;      (* region to hold current highlighted cells *)
10 109 --
10 110 --      tmpRgn := MakeNewRgn;
10 111 --      fTmpRgn := tmpRgn;      (* temporary region for use on-the-fly *)
10 112 --
10 113 --      tmpRgn := MakeNewRgn;
10 114 --      fTmpSelRgn := tmpRgn;      (* temporary region for selection *)
10 115 --
10 116 --      tmpRgn := MakeNewRgn;
10 117 --      fTmpHLRgn := tmpRgn;      (* temporary region for highlighting *)
10 118 --
10 119 --      fSingleSelection := singleSelection;
10 120 --      fWholeRect := FALSE;      (* the new selected region is not a rectangle *)
10 121 --      fLastWholeRect := FALSE;      (* the old selected region is not a rectangle *)
10 122 --      fAllowDimSelection := FALSE; (* dimming the selection is possible *)
10 123 --
10 124 --      IF (numOfRows > 0) THEN
10 125 --          InsRowFirst (numOfRows, rowHeight);
10 126 --      IF (numOfCols > 0) THEN
10 127 --          InsColFirst (numOfCols, colWidth);
10 128 --
10 129 -- A  END; (* TGridView.IGridView *)
10 130 --      ($ENDC qTemplateViews)
10 131 --
10 132 --
10 133 --
10 134 --      {*****}
10 135 --      (* TGridView.IRes *)
10 136 --      (* Initialize the GridView for use with view templates. *)
10 137 --      {*****}
10 138 --      ($IFC qTemplateViews)
10 139 --      ($S GVOpen)
10 140 -- A  PROCEDURE TGridView.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
10 141 --      VAR
10 142 --          aHRunArray:      HRunArray;
10 143 --          tmpRgn:          RgnHandle;
10 144 --
10 145 -- A  BEGIN
10 146 --      INHERITED IRes (itsDocument, itsSuperview, itsParams);
10 147 --
10 148 --      WITH GridViewTemplatePtr (itsParams)^ DO
10 149 --      BEGIN
10 150 --          fNumOfRows := 0;
10 151 --          fNumOfCols := 0;
10 152 --
10 153 --          fAdornRows := adornRows;
10 154 --          fAdornCols := adornCols;
10 155 --
10 156 --          (* Make sure the insets are evenly divided between top/bottom or left/right *)
10 157 --          IF ODD (rowInset) THEN
10 158 --              fRowInset := rowInset + 1
10 159 --          ELSE
10 160 --              fRowInset := rowInset;
10 161 --
10 162 --          IF ODD (colInset) THEN
10 163 --              fColInset := colInset + 1
10 164 --          ELSE
10 165 --              fColInset := colInset;
10 166 --
10 167 --
10 168 --          fSingleSelection := singleSelection;
10 169 --          fWholeRect := FALSE;      (* the new selected region is not a rectangle *)
10 170 --          fLastWholeRect := FALSE;      (* the old selected region is not a rectangle *)
10 171 --          fAllowDimSelection := FALSE; (* dimming the selection is possible *)
10 172 --
10 173 --
10 174 --          (* "Munger" is used to grow and shrink the array containing row heights and
10 175 --          * column widths. The fRowHeights and fColWidths are initially handles
10 176 --          * to empty arrays.
10 177 --          *)
10 178 --
10 179 --          aHRunArray := HRunArray (NewPermHandle (0));
10 180 --          FailNil (aHRunArray);
10 181 --          fColWidths := aHRunArray;
10 182 --
10 183 --          aHRunArray := HRunArray (NewPermHandle (0));
10 184 --          FailNil (aHRunArray);
10 185 --          fRowHeights := aHRunArray;
10 186 --
10 187 --
10 188 --          (* for cacheing run array values *)
10 189 --          fLastRow := 0;
10 190 --          fLastRowChunkNum := 0;
10 191 --          fLastRowTotal := 0;
10 192 --          fLastRowIndex := 1;
10 193 --          fNumOfRowChunks := 0;
10 194 --
10 195 --          fLastCol := 0;
10 196 --          fLastColChunkNum := 0;
10 197 --          fLastColTotal := 0;

```

```

10 198 --      fLastColIndex := 1;
10 199 --      fNumOfColChunks := 0;
10 200 --
10 201 --      fMaxHorizontal := 0;
10 202 --      fMaxVertical := 0;
10 203 --
10 204 --      tmpRgn := MakeNewRgn;
10 205 --      fSelections := tmpRgn;      (* region to hold current selections *)
10 206 --
10 207 --      tmpRgn := MakeNewRgn;
10 208 --      fHLRegion := tmpRgn;      (* region to hold current highlighted cells *)
10 209 --
10 210 --      tmpRgn := MakeNewRgn;
10 211 --      fTmpRgn := tmpRgn;      (* temporary region for use on-the-fly *)
10 212 --
10 213 --      tmpRgn := MakeNewRgn;
10 214 --      fTmpSelRgn := tmpRgn;      (* temporary region for selection *)
10 215 --
10 216 --      tmpRgn := MakeNewRgn;
10 217 --      fTmpHLRgn := tmpRgn;      (* temporary region for highlighting *)
10 218 --
10 219 --      IF (itsSuperView <> NIL) & (fSizeDeterminer[h] <> sizeFixed) THEN
10 220 --          fSize.h := itsSuperView.fSize.h;
10 221 --
10 222 --      IF (numOfRows > 0) THEN
10 223 --          InsRowFirst (numOfRows, rowHeight);
10 224 --      IF (numOfCols > 0) THEN
10 225 --          InsColFirst (numOfCols, colWidth);
10 226 --
10 227 --      END;
10 228 --
10 229 --      OffsetPtr (itsParams, SIZEOF (GridViewTemplate));
10 230 -- A  END; (* TGridView.IRes *)
10 231 --      {$ENDC qTemplateViews}
10 232 --
10 233 --
10 234 --      {*****}
10 235 --      (* TGridView.WRes *)
10 236 --      (* Write the resource file. *)
10 237 --      {*****}
10 238 --      {$IFC qWriteTemplates}
10 239 --      {$S GVNonRes}
10 240 -- A  PROCEDURE TGridView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
10 241 --
10 242 --      VAR
10 243 --          gvPtr:      GridViewTemplatePtr;
10 244 --
10 245 -- A  BEGIN
10 246 --      INHERITED WRes(theResource, itsParams);
10 247 --
10 248 --      gvPtr := GridViewTemplatePtr(ExpandPtr(theResource, itsParams,
10 249 --          SIZEOF(GridViewTemplate)));
10 250 --
10 251 --      WITH gvPtr^ DO
10 252 --      BEGIN
10 253 --          numOfRows := fNumOfRows;
10 254 --          numOfCols := fNumOfCols;
10 255 --          IF fNumOfRows > 0 THEN
10 256 --              rowHeight := GetRowHeight(1)
10 257 --          ELSE
10 258 --              rowHeight := 0;
10 259 --          IF fNumOfCols > 0 THEN
10 260 --              colWidth := GetColWidth(1)
10 261 --          ELSE
10 262 --              colWidth := 0;
10 263 --          rowInset := fRowInset;
10 264 --          colInset := fColInset;
10 265 --          adornRows := fAdornRows;
10 266 --          adornCols := fAdornCols;
10 267 --          singleSelection := fSingleSelection;
10 268 --      END;
10 269 -- A  END; (* TGridView.WRes *)
10 270 --
10 271 --
10 272 --      {*****}
10 273 --      (* TGridView.WriteRes *)
10 274 --      (* Set up for writing the resource file *)
10 275 --      {*****}
10 276 --      {$S GVNonRes}
10 277 -- A  PROCEDURE TGridView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
10 278 -- A  BEGIN
10 279 --      gWResSignature := 'grid'; gWResType := 'TGridView';
10 280 --      WRes(theResource, itsParams);
10 281 -- A  END; (* TGridView.WriteRes *)
10 282 --      {$ENDC qWriteTemplates}
10 283 --
10 284 --
10 285 --      {*****}
10 286 --      (* TGridView.Free *)
10 287 --      (* Free the TGridView object and any associated objects *)
10 288 --      {*****}
10 289 --      {$S GVClose}
10 290 -- A  PROCEDURE TGridView.Free; OVERRIDE;
10 291 -- A  BEGIN
10 292 --      DisposeHandle (fSelections);      (* Dispose regions *)
10 293 --      DisposeHandle (fHLRegion);
10 294 --      DisposeHandle (fTmpRgn);
10 295 --      DisposeHandle (fTmpSelRgn);
10 296 --      DisposeHandle (fTmpHLRgn);

```

```

10 297 --      DisposIfHandle (fColWidths);      (* Dispose run arrays *)
10 298 --      DisposIfHandle (fRowHeights);
10 299 --
10 300 --      INHERITED Free;
10 301 -0 A      END; (* TGridView.Free *)
10 302 --
10 303 --
10 304 --      {*****}
10 305 --      (* TGridView.AllCellsDo *)
10 306 --      (* For all cells in the view perform "DoToCell". *)
10 307 --      {*****}
10 308 --      {$S GVRes}
10 309 -- A      PROCEDURE TGridView.AllCellsDo (PROCEDURE DoToCell (aCell: GridCell));
10 310 --      VAR
10 311 --          startCell: GridCell;
10 312 --          stopCell: GridCell;
10 313 --
10 314 0- A      BEGIN
10 315 --          startCell.h := 1;
10 316 --          startCell.v := 1;
10 317 --          stopCell.h := fNumOfCols;
10 318 --          stopCell.v := fNumOfRows;
10 319 --
10 320 --          EachCellDo (startCell, stopCell, DoToCell);
10 321 -0 A      END; (* TGridView.AllCellsDo *)
10 322 --
10 323 --
10 324 --      {*****}
10 325 --      (* TGridView.AdornCol *)
10 326 --      (* Draw the column adornments for the whole column. *)
10 327 --      {*****}
10 328 --      {$S GVRes}
10 329 -- A      PROCEDURE TGridView.AdornCol (aCol: INTEGER; area: Rect);
10 330 0- A      BEGIN
10 331 -0 A      END; (* TGridView.AdornCol *)
10 332 --
10 333 --
10 334 --      {*****}
10 335 --      (* TGridView.AdornRow *)
10 336 --      (* Draw the row adornments for the whole row. *)
10 337 --      {*****}
10 338 --      {$S GVRes}
10 339 -- A      PROCEDURE TGridView.AdornRow (aRow: INTEGER; area: Rect);
10 340 0- A      BEGIN
10 341 -0 A      END; (* TGridView.AdornRow *)
10 342 --
10 343 --
10 344 --      {*****}
10 345 --      (* TGridView.CalcMinSize *)
10 346 --      (* This method adjusts the size of the extent by the amount of space *)
10 347 --      (* needed to display all the rows and columns. It is called (by TView. *)
10 348 --      (* ComputeExtent) whenever a row or column is added or deleted. *)
10 349 --      {*****}
10 350 --      {$S GVRes}
10 351 -- A      PROCEDURE TGridView.CalcMinSize (VAR minSize: VPoint); OVERRIDE;
10 352 0- A      BEGIN
10 353 --          INHERITED CalcMinSize (minSize);
10 354 --
10 355 --          (* Set the amount of room needed for that many items *)
10 356 --          minSize.v := fMaxVertical;
10 357 --          minSize.h := fMaxHorizontal;
10 358 -0 A      END; (* TGridView.CalcMinSize *)
10 359 --
10 360 --
10 361 --      {*****}
10 362 --      (* TGridView.CanSelectCell *)
10 363 --      (* Determine if a cell can be selected. By default, this method returns *)
10 364 --      (* TRUE for everything. *)
10 365 --      {*****}
10 366 --      {$S GVRes}
10 367 -- A      FUNCTION TGridView.CanSelectCell (aCell: GridCell): BOOLEAN;
10 368 --
10 369 0- A      BEGIN
10 370 --          {For now, allow any item to be selected.}
10 371 --          CanSelectCell := TRUE;
10 372 -0 A      END; (* TGridView.CanSelectCell *)
10 373 --
10 374 --
10 375 --      {*****}
10 376 --      (* TGridView.CellExists *)
10 377 --      (* Determine if a cell can be drawn. By default, this method returns *)
10 378 --      (* TRUE for everything. *)
10 379 --      {*****}
10 380 --      {$S GVRes}
10 381 -- A      FUNCTION TGridView.CellExists (aCell: GridCell): BOOLEAN;
10 382 --
10 383 0- A      BEGIN
10 384 --          {allow any cell to be drawn.}
10 385 --          CellExists := TRUE;
10 386 -0 A      END; (* TGridView.CellExists *)
10 387 --
10 388 --
10 389 --      {*****}
10 390 --      (* TGridView.CellToVRect *)
10 391 --      (* Calculate the bounding rectangle for a given cell. This includes *)
10 392 --      (* any row or column insets. *)
10 393 --      {*****}
10 394 --      {$S GVRes}
10 395 -- A      PROCEDURE TGridView.CellToVRect (aCell: GridCell; VAR aRect: VRect);

```

```

10 396 -- VAR
10 397 --     bRect:      VRect;
10 398 --
10 399 0- A BEGIN
10 400 --     IF (aCell.h < 1) | (aCell.v < 1) |
10 401 --         (aCell.h > fNumOfCols) | (aCell.v > fNumOfRows) THEN
10 402 1- BEGIN
10 403 --     ($IFC qRangeCheck)
10 404 --     ($IFC qDebug)
10 405 --         Writeln('aCell.h = ', aCell.h:6, ' fNumOfCols = ', fNumOfCols:6);
10 406 --         Writeln('aCell.v = ', aCell.v:6, ' fNumOfRows = ', fNumOfRows:6);
10 407 --     ($ENDC)
10 408 --         ProgramBreak('Range Check in CellToVRect');
10 409 --     ($ENDC)
10 410 --         SetVRect (aRect, 0, 0, 0, 0);
10 411 1- END
10 412 --     ELSE (* all the params look OK *)
10 413 1- BEGIN
10 414 --         RowToVRect (aCell.v, 1, aRect);
10 415 --         ColToVRect (aCell.h, 1, bRect);
10 416 --         WITH aRect DO
10 417 2- BEGIN
10 418 --             left := bRect.left;
10 419 --             right := bRect.right;
10 420 2- END;
10 421 1- END;
10 422 0- A END; (* TGridView.CellToVRect *)
10 423 --
10 424 --
10 425 -- {*****}
10 426 -- { * TGridView.ColToVRect *}
10 427 -- { * Calculate the bounding rectangle for the given column(s). The top and *}
10 428 -- { * bottom of the rectangle are the same as the view. *}
10 429 -- { * The rectangle includes the cells in that column and the column inset. *}
10 430 -- {*****}
10 431 -- {$S GVRes}
10 432 0- A PROCEDURE TGridView.ColToVRect (aCol: INTEGER; numOfCols: INTEGER; VAR aRect: VRect);
10 433 -- VAR
10 434 --     width:      LONGINT;
10 435 --     Num:        INTEGER;
10 436 --     theTotal:   LONGINT;
10 437 --     Index:      INTEGER;
10 438 --     i:          INTEGER;
10 439 --
10 440 0- A BEGIN
10 441 --     IF (aCol < 1) | (numOfCols < 1) | (aCol + numOfCols - 1 > fNumOfCols) THEN
10 442 1- BEGIN
10 443 --     ($IFC qRangeCheck)
10 444 --     ($IFC qDebug)
10 445 --         Writeln('fNumOfCols = ', fNumOfCols:1, ' aCol = ', aCol:1);
10 446 --     ($ENDC)
10 447 --         ProgramBreak('Range Check in ColToVRect');
10 448 --     ($ENDC)
10 449 --         SetVRect (aRect, 0, 0, 0, 0);
10 450 1- END
10 451 --     ELSE (* all the params look OK *)
10 452 1- BEGIN
10 453 --         width := 0;
10 454 --
10 455 --         IF (fNumOfColChunks = 1) THEN (* only one column height *)
10 456 2- BEGIN
10 457 --             num := 0;
10 458 --             theTotal := IntMultiply (fColWidths^[num].size, aCol-1);
10 459 --             width := IntMultiply (numOfCols, fColWidths^[num].size);
10 460 2- END
10 461 --         ELSE
10 462 2- BEGIN
10 463 --             IF FindColChunk (aCol, Num, theTotal, Index) THEN
10 464 3- BEGIN
10 465 --                 theTotal := theTotal + IntMultiply ((Index-1), fColWidths^[Num].size);
10 466 --                 For i := aCol TO (aCol + numOfCols - 1) DO
10 467 --                     width := width + GetColWidth (i);
10 468 3- END;
10 469 2- END;
10 470 --
10 471 --         WITH aRect DO
10 472 2- BEGIN
10 473 --             top := 0;
10 474 --             left := theTotal;
10 475 --             bottom := fMaxVertical;
10 476 --             right := theTotal + width;
10 477 2- END;
10 478 1- END;
10 479 0- A END; (* TGridView.ColToVRect *)
10 480 --
10 481 --
10 482 -- {*****}
10 483 -- { * TGridView.CreateHighlightRgn *}
10 484 -- { * This guy goes through all the cells in oldRgn and creates a *}
10 485 -- { * newRgn to be used for highlighting. *}
10 486 -- {*****}
10 487 -- {$S GVRes}
10 488 0- A PROCEDURE TGridView.CreateHighlightRgn (oldRgn, newRgn: RgnHandle; wholeRect: BOOLEAN);
10 489 -- VAR
10 490 --     aRect:      VRect;
10 491 --     bRect:      VRect;
10 492 --     aQDRect:    Rect;
10 493 --
10 494 0- B PROCEDURE AddToHLRegion (aCell: GridCell);

```

```

10 495 0- B      BEGIN
10 496 --      IF ((aCell.h > 0) & (aCell.v > 0) &
10 497 --          (aCell.h <= fNumOfCols) & (aCell.v <= fNumOfRows)) THEN
10 498 1-          BEGIN
10 499 --              CellToVRect (aCell, aRect);
10 500 --              ViewToQDRect (aRect, aQDRect);
10 501 --              RectRgn (fTmpHLRgn, aQDRect);
10 502 --              UnionRgn (fTmpHLRgn, newRgn, newRgn);
10 503 -1          END;
10 504 -0 B      END;
10 505 --
10 506 0- A      BEGIN
10 507 --      IF EmptyRgn (oldRgn) THEN
10 508 --          SetEmptyRgn (newRgn)
10 509 --
10 510 --      ELSE IF wholeRect THEN
10 511 1-          BEGIN
10 512 --              aQDRect := oldRgn^.rgnBBox;
10 513 --              aQDRect.bottom := aQDRect.bottom - 1;
10 514 --              aQDRect.right := aQDRect.right - 1;
10 515 --
10 516 --              IF ((aQDRect.top > 0) & (aQDRect.left > 0) &
10 517 --                  (aQDRect.right <= fNumOfCols) &
10 518 --                  (aQDRect.bottom <= fNumOfRows) &
10 519 --                  (aQDRect.top <= aQDRect.bottom) &
10 520 --                  (aQDRect.left <= aQDRect.right)) THEN
10 521 2-          BEGIN
10 522 --              CellToVRect (aQDRect.topleft, aRect);
10 523 --              CellToVRect (aQDRect.botright, bRect);
10 524 --              aRect.botright := bRect.botright;
10 525 --              ViewToQDRect (aRect, aQDRect);
10 526 --              RectRgn (newRgn, aQDRect);
10 527 -2          END;
10 528 -1          END
10 529 --      ELSE
10 530 1-          BEGIN
10 531 --              SetEmptyRgn (newRgn);
10 532 --              EachInRgn (oldRgn, AddToHLRegion);
10 533 -1          END;
10 534 -0 A      END; (* TGridView.CreateHighlightRgn *)
10 535 --
10 536 --
10 537 --      {*****}
10 538 --      (* TGridView.DoHighlightSelection *)
10 539 --      (* Figure out what needs to be highlighted and call DoHilite. *)
10 540 --      {*****}
10 541 --      {$S GVRes}
10 542 -- A      PROCEDURE TGridView.DoHighlightSelection (fromHL, toHL: HLState); OVERRIDE;
10 543 --      VAR
10 544 --          hlRect:    Rect;
10 545 --          aQDRect:   Rect;
10 546 --          aRect:     VRect;
10 547 --          bRect:     VRect;
10 548 --
10 549 0- A      BEGIN
10 550 --      IF NOT EmptyRgn (fHLRegion) THEN
10 551 1-          BEGIN
10 552 --          CreateHighlightRgn (fHLRegion, fTmpRgn, fWholeRect);
10 553 --          DoHilite (fTmpRgn, fromHL, toHL)
10 554 -1          END;
10 555 -0 A      END; (* TGridView.DoHighlightSelection *)
10 556 --
10 557 --
10 558 --      {*****}
10 559 --      (* TGridView.DoHilite *)
10 560 --      (* Invert the cell. Override this guy to have different highlighting. *)
10 561 --      {*****}
10 562 --      {$S GVRes}
10 563 -- A      PROCEDURE TGridView.DoHilite (hlRgn: RgnHandle; fromHL, toHL: HLState);
10 564 --      VAR
10 565 --          aRect:     VRect;
10 566 --          bRect:     VRect;
10 567 --          aQDRect:   Rect;
10 568 --          bQDRect:   Rect;
10 569 0- A      BEGIN
10 570 --      IF focus THEN
10 571 1-          BEGIN
10 572 --              IF NOT followDimSelection THEN
10 573 2-          BEGIN
10 574 --              IF fromHL = hldim THEN
10 575 --                  fromHL := hloff;
10 576 --              IF toHL = hldim THEN
10 577 --                  toHL := hloff;
10 578 -2          END;
10 579 --
10 580 --          PenNormal;
10 581 --          IF (fromHL + toHL = hlonoff) THEN
10 582 2-          BEGIN
10 583 --              InvertRgn (hlRgn); (* highlight the cells *)
10 584 -2          END
10 585 --          ELSE IF fromHL = hldim THEN
10 586 2-          BEGIN
10 587 --              IF toHL = hlon THEN
10 588 3-          BEGIN
10 589 --                  InvertRgn (hlRgn);
10 590 --                  PenMode (patXor);
10 591 --                  FrameRgn (hlRgn); (* erase the 'dim' *)
10 592 -3          END
10 593 --          ELSE (* toHL = hloff *)

```

```

10 594 3-      BEGIN
10 595 --      PenMode (patXor);
10 596 --      FrameRgn (hlRgn);    (* erase the 'dim' *)
10 597 -3      END;
10 598 -2      END
10 599 --      ELSE IF toHL = hlDim THEN
10 600 2-      BEGIN
10 601 --      IF fromHL = hlOn THEN
10 602 --      InvertRgn (hlRgn);
10 603 --      PenMode (patXor);
10 604 --      FrameRgn (hlRgn);    (* 'dim' the rect *)
10 605 -2      END;
10 606 -1      END;
10 607 -0 A  END; (* TGridView.DoHilite *)
10 608 --
10 609 --
10 610 --      {*****}
10 611 --      (* TGridView.DoMouseCommand *)
10 612 --      (* Determine if the mouse click was in a cell, a row, a column, or a *)
10 613 --      (* vertex. (A vertex is the place where a row and column meet.) Override *)
10 614 --      (* IdentifyPoint to change the logic for how a point gets classified. *)
10 615 --      (* Based on the point classification, a command object is created and *)
10 616 --      (* control is passed on to that object. *)
10 617 --      {*****}
10 618 --      {$$ GVRes}
10 619 -- A  FUNCTION TGridView.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
10 620 --      VAR hysteresis: Point) : TCommand; OVERRIDE;
10 621 --
10 622 --      VAR
10 623 --      whichPart:      GridViewPart;
10 624 --      aRow, aCol:      INTEGER;
10 625 --      aCellSelectCommand: TCellSelectCommand;
10 626 --      aRowSelectCommand: TRowSelectCommand;
10 627 --      aColumnSelectCommand: TColSelectCommand;
10 628 --      aVertexSelectCommand: TVertexSelectCommand;
10 629 --      aCell:      GridCell;
10 630 --
10 631 0- A  BEGIN
10 632 --
10 633 --      whichPart := IdentifyPoint (themouse, aRow, aCol);
10 634 --      aCell.h := aCol;
10 635 --      aCell.v := aRow;
10 636 --
10 637 1-      CASE whichPart OF
10 638 --      inCell:
10 639 2-      BEGIN
10 640 --      NEW (aCellSelectCommand);
10 641 --      FailNIL (aCellSelectCommand);
10 642 --      aCellSelectCommand.ICellSelectCommand (SELF, aCell, info.theShiftKey,
10 643 --      info.theCmdKey);
10 644 --      DoMouseCommand := aCellSelectCommand;
10 645 -2      END;
10 646 --
10 647 --      inRow:
10 648 2-      BEGIN
10 649 --      NEW (aRowSelectCommand);
10 650 --      FailNIL (aRowSelectCommand);
10 651 --      aRowSelectCommand.IRowSelectCommand (SELF, aRow, info.theShiftKey,
10 652 --      info.theCmdKey);
10 653 --      DoMouseCommand := aRowSelectCommand;
10 654 -2      END;
10 655 --
10 656 --      inColumn:
10 657 2-      BEGIN
10 658 --      NEW (aColumnSelectCommand);
10 659 --      FailNIL (aColumnSelectCommand);
10 660 --      aColumnSelectCommand.IColSelectCommand (SELF, aCol, info.theShiftKey,
10 661 --      info.theCmdKey);
10 662 --      DoMouseCommand := aColumnSelectCommand;
10 663 -2      END;
10 664 --
10 665 --      inVertex:
10 666 2-      BEGIN
10 667 --      NEW (aVertexSelectCommand);
10 668 --      FailNIL (aVertexSelectCommand);
10 669 --      aVertexSelectCommand.IVertexSelectCommand (SELF, aRow, aCol,
10 670 --      info.theShiftKey, info.theCmdKey);
10 671 --      DoMouseCommand := aVertexSelectCommand;
10 672 -2      END;
10 673 --
10 674 --      OTHERWISE
10 675 --      DoMouseCommand := gNoChanges;
10 676 -1      END;
10 677 -0 A  END; (* TGridView.DoMouseCommand *)
10 678 --
10 679 --
10 680 --      {*****}
10 681 --      (* TGridView.Draw *)
10 682 --      (* Draw each cell in "area". If either adorn is TRUE then call that *)
10 683 --      (* Adorn. ONLY the cells that need to be re-drawn are drawn. *)
10 684 --      {*****}
10 685 --      {$$ GVRes}
10 686 -- A  PROCEDURE TGridView.Draw (area: Rect); OVERRIDE;
10 687 --
10 688 --      VAR
10 689 --      aRect:      VRect;
10 690 --      bRect:      VRect;
10 691 --      aQDRect:      Rect;
10 692 --      i:      INTEGER;

```

```

10 693 --      startCell:      GridCell;
10 694 --      stopCell:       GridCell;
10 695 --      viewArea:       VRect;
10 696 --      colWidth:       INTEGER;
10 697 --      rowHeight:      INTEGER;
10 698 --
10 699 0- A BEGIN
10 700 --      IF (fNumOfRows > 0) & (fNumOfCols > 0) THEN
10 701 1-      BEGIN (* make sure we have something to draw *)
10 702 --          QDtoViewRect (area, viewArea);
10 703 --          viewArea.right := viewArea.right - 1;
10 704 --          viewArea.bottom := viewArea.bottom - 1;
10 705 --
10 706 --
10 707 --          WITH viewArea DO
10 708 2-          BEGIN
10 709 --              startCell := VPointToCell (topleft);
10 710 --              stopCell := VPointToCell (botright);
10 711 -2          END;
10 712 --
10 713 --
10 714 --          CellToVRect (startCell, aRect);
10 715 --          CellToVRect (stopCell, bRect);
10 716 --          aRect.botright := bRect.botright;
10 717 --          ViewToQDRect (aRect, area);
10 718 --
10 719 --          DrawRangeOfCells (startCell, stopCell, area);
10 720 --
10 721 --          IF fAdornCols THEN
10 722 2-          BEGIN
10 723 --              aQDRect := area;
10 724 --
10 725 --              IF fNumOfColChunks = 1 THEN (* only one width *)
10 726 --                  colWidth := fColWidths^^[0].size;
10 727 --
10 728 --              For i := startCell.h to stopCell.h DO
10 729 3-              BEGIN
10 730 --                  IF fNumOfColChunks = 1 THEN (* only one height *)
10 731 --                      aQDRect.right := aQDRect.left + colWidth
10 732 --                  ELSE
10 733 --                      aQDRect.right := aQDRect.left + GetColWidth (i);
10 734 --
10 735 --                      AdornCol (i, aQDRect);
10 736 --                      aQDRect.left := aQDRect.right;
10 737 -3              END;
10 738 -2          END;
10 739 --
10 740 --          IF fAdornRows THEN
10 741 2-          BEGIN
10 742 --              aQDRect := area;
10 743 --
10 744 --              IF fNumOfRowChunks = 1 THEN (* only one height *)
10 745 --                  rowHeight := fRowHeights^^[0].size;
10 746 --
10 747 --              For i := startCell.v to stopCell.v DO
10 748 3-              BEGIN
10 749 --                  IF fNumOfRowChunks = 1 THEN (* only one height *)
10 750 --                      aQDRect.bottom := aQDRect.top + rowheight
10 751 --                  ELSE
10 752 --                      aQDRect.bottom := aQDRect.top + GetRowHeight (i);
10 753 --
10 754 --                      AdornRow (i, aQDRect);
10 755 --                      aQDRect.top := aQDRect.bottom;
10 756 -3              END;
10 757 -2          END;
10 758 -1          END;
10 759 -0 A END; (* TGridView.Draw *)
10 760 --
10 761 --
10 762 --      {*****}
10 763 --      (* TGridView.DrawRangeOfCells *)
10 764 --      (* This method calls DrawCell for each cell that needs to be drawn. *)
10 765 --      {*****}
10 766 --      {SS GVRes}
10 767 -- A PROCEDURE TGridView.DrawRangeOfCells (startCell: GridCell; aQDRect: Rect);
10 768 --      VAR
10 769 --          colWidth:  INTEGER;
10 770 --          rowHeight: INTEGER;
10 771 --          i, j:      INTEGER;
10 772 --          aCell:     GridCell;
10 773 --          left:      INTEGER;
10 774 --
10 775 0- A BEGIN
10 776 --
10 777 --          aQDRect.left := aQDRect.left + BSR (fColInset, 1); (* fColInset DIV 2 *)
10 778 --          aQDRect.top := aQDRect.top + BSR (fRowInset, 1); (* fRowInset DIV 2 *)
10 779 --          left := aQDRect.left;
10 780 --
10 781 --          IF fNumOfColChunks = 1 THEN (* only one width *)
10 782 --              colWidth := GetColWidth (1);
10 783 --          IF fNumOfRowChunks = 1 THEN (* only one height *)
10 784 --              rowHeight := GetRowHeight (1);
10 785 --
10 786 --
10 787 --          For j := startCell.v to stopCell.v DO
10 788 1-          BEGIN
10 789 --              IF fNumOfRowChunks = 1 THEN (* only one height *)
10 790 --                  aQDRect.bottom := aQDRect.top + rowheight - fRowInset
10 791 --              ELSE

```



```

10 792 --      aQDRect.bottom := aQDRect.top + GetRowHeight (j) - fRowInset;
10 793 --
10 794 --      aQDRect.left := left; (* start back at the left for the next row *)
10 795 --
10 796 --      For i := startCell.h to stopCell.h DO
10 797 2- BEGIN
10 798 --          IF fNumOfColChunks = 1 THEN (* only one height *)
10 799 --              aQDRect.right := aQDRect.left + colWidth - fColInset
10 800 --          ELSE
10 801 --              aQDRect.right := aQDRect.left + GetColWidth (i) - fColInset;
10 802 --
10 803 --
10 804 --              aCell.h := i;
10 805 --              aCell.v := j;
10 806 --              IF CellExists (aCell) THEN
10 807 --                  DrawCell (aCell, aQDRect);
10 808 --
10 809 --              aQDRect.left := aQDRect.right + fColInset;
10 810 2-          END;
10 811 --          aQDRect.top := aQDRect.bottom + fRowInset;
10 812 1-      END;
10 813 0 A  END; (* TGridView.DrawRangeOfCells *)
10 814 --
10 815 --
10 816 --      {*****}
10 817 --      (* TGridView.DrawCell *)
10 818 --      (* This method draws the given cell inside the specified rectangle. The *)
10 819 --      (* rectangle is the cell size without any "insets". *)
10 820 --      (* It MUST be overridden!! *)
10 821 --      {*****}
10 822 --      ($$ GVRes)
10 823 0 A  PROCEDURE TGridView.DrawCell (aCell: GridCell; aQDRect: Rect);
10 824 0- A  BEGIN
10 825 --          (* Should always be overridden. *)
10 826 --
10 827 --          ($IFC qDebug)
10 828 --          WRITELN ('TGridView: DrawCell MUST be overridden!');
10 829 --          ($ENDC qDebug)
10 830 0 A  END; (* TGridView.DrawCell *)
10 831 --
10 832 --
10 833 --      {*****}
10 834 --      (* TGridView.DelColAt *)
10 835 --      (* Delete numOfCols starting at aCol. This method causes a re-calculation *)
10 836 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 837 --      {*****}
10 838 --      ($$ GVNonRes)
10 839 0 A  PROCEDURE TGridView.DelColAt (aCol: INTEGER; numOfCols: INTEGER);
10 840 --      VAR
10 841 --          offset:    LONGINT;
10 842 --          dummy:     Ptr;
10 843 --          result:    INTEGER;
10 844 --          i:         INTEGER;
10 845 --          aRect:     VRect;
10 846 --          Num:       INTEGER;
10 847 --          theTotal:  LONGINT;
10 848 --          Index:     INTEGER;
10 849 --          chunkFound: BOOLEAN;
10 850 --          selRect:   Rect;
10 851 --
10 852 0- A  BEGIN
10 853 --          IF (aCol < 1) | (numOfCols < 1) | (aCol + numOfCols - 1 > fNumOfCols) THEN
10 854 1-      BEGIN
10 855 --          IF numOfCols <> 0 THEN
10 856 2-      BEGIN
10 857 --          ($IFC qRangeCheck)
10 858 --          ($IFC qDebug)
10 859 --          Writeln('fNumOfCols = ', fNumOfCols:1, ' aCol = ', aCol:1);
10 860 --          ($ENDC)
10 861 --          ProgramBreak('Range Check in DelColAt');
10 862 --          ($ENDC)
10 863 2-      END;
10 864 1-      END
10 865 --      ELSE
10 866 1-      BEGIN
10 867 --          (* make sure we take any deleted items out of the selection *)
10 868 --          selRect.top := 1;
10 869 --          selRect.left := aCol;
10 870 --          selRect.bottom := fNumOfRows + 1;
10 871 --          selRect.right := aCol + numOfCols;
10 872 --          RectRgn (fTmpRgn, selRect);
10 873 --          DiffRgn (fSelections, fTmpRgn, fSelections);
10 874 --          DiffRgn (fHLRegion, fTmpRgn, fHLRegion);
10 875 --
10 876 --          ColToVRect (Max (1, aCol), Max (1, fNumOfCols - aCol), aRect);
10 877 --          chunkFound := FindColChunk (aCol, Num, theTotal, Index);
10 878 --
10 879 --          FOR i := 1 TO numOfCols DO
10 880 2-      BEGIN
10 881 --          fMaxHorizontal := fMaxHorizontal - fColWidths^[Num].size;
10 882 --
10 883 --          fColWidths^[Num].count := fColWidths^[Num].count - 1;
10 884 --
10 885 --          IF (fColWidths^[Num].count < Index) THEN
10 886 3-      BEGIN
10 887 --          IF (fColWidths^[Num].count = 0) THEN
10 888 4-      BEGIN
10 889 --          (* need to delete that chunk *)
10 890 --          offset := LONGINT (IntMultiply (4, Num));

```

```

10 891 --      result := Munger (Handle (fColWidths), offset, Ptr (Ord (fColWidths^) + offset),
10 892 --      sizeof (LONGINT), dummy, 0);
10 893 --      fNumOfColChunks := fNumOfColChunks - 1;
10 894 --
10 895 --      (* see if we can consolidate chunks *)
10 896 --      IF (Num > 0) & (fColWidths^[Num - 1].size = fColWidths^[Num].size) THEN
10 897 5-      BEGIN
10 898 --          fColWidths^[Num - 1].count := fColWidths^[Num - 1].count +
10 899 --          fColWidths^[Num].count;
10 900 --          (* need to delete that chunk *)
10 901 --          offset := LONGINT (IntMultiply (4, Num));
10 902 --          result := Munger (Handle (fColWidths), offset, Ptr (Ord (fColWidths^) + offset),
10 903 --          sizeof (LONGINT), dummy, 0);
10 904 --          fNumOfColChunks := fNumOfColChunks - 1;
10 905 5-      END;
10 906 4-      END
10 907 --      ELSE
10 908 4-      BEGIN
10 909 --          Num := Num + 1;
10 910 4-      END;
10 911 --      Index := 1;
10 912 3-      END;
10 913 2-      END;
10 914 --
10 915 --      fNumOfCols := fNumOfCols - numOfCols;
10 916 --
10 917 --      (* reset the run array info *)
10 918 --      fLastCol := 0;
10 919 --      fLastColChunkNum := 0;
10 920 --      fLastColTotal := 0;
10 921 --      fLastColIndex := 1;
10 922 --
10 923 --      AdjustSize;
10 924 --
10 925 --      InvalidateVRect (aRect);
10 926 1-      END;
10 927 0 A  END; (* TGridView.DelColAt *)
10 928 --
10 929 --
10 930 --      {*****}
10 931 --      (* TGridView.DelRowAt *)
10 932 --      (* Delete numOfRows starting at aRow. This method causes a re-calculation *)
10 933 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 934 --      {*****}
10 935 --      {$S GVNonRes}
10 936 -- A  PROCEDURE TGridView.DelRowAt (aRow: INTEGER; numOfRows: INTEGER);
10 937 --      VAR
10 938 --          offset: LONGINT;
10 939 --          dummy: Ptr;
10 940 --          result: INTEGER;
10 941 --          i: INTEGER;
10 942 --          aRect: VRect;
10 943 --          Num: INTEGER;
10 944 --          theTotal: LONGINT;
10 945 --          Index: INTEGER;
10 946 --          chunkFound: BOOLEAN;
10 947 --          selRect: Rect;
10 948 --
10 949 0- A  BEGIN
10 950 --      IF (aRow < 1) | (numOfRows < 1) | (aRow + numOfRows - 1 > fNumOfRows) THEN
10 951 1-      BEGIN
10 952 --          IF numOfRows <> 0 THEN
10 953 2-      BEGIN
10 954 --          ($IFC qRangeCheck)
10 955 --          ($IFC qDebug)
10 956 --          Writeln('fNumOfRows = ', fNumOfRows:1, ' aRow = ', aRow:1);
10 957 --          ($ENDC)
10 958 --          ProgramBreak('Range Check in DelRowAt');
10 959 --          ($ENDC)
10 960 2-      END;
10 961 1-      END
10 962 --      ELSE
10 963 1-      BEGIN
10 964 --          (* make sure we take any deleted items out of the selection *)
10 965 --          selRect.top := aRow;
10 966 --          selRect.left := 1;
10 967 --          selRect.bottom := aRow + numOfRows;
10 968 --          selRect.right := fNumOfCols + 1;
10 969 --          RectRgn (fTmpRgn, selRect);
10 970 --          DiffRgn (fSelections, fTmpRgn, fSelections);
10 971 --          DiffRgn (fHLRegion, fTmpRgn, fHLRegion);
10 972 --
10 973 --          RowToVRect (Max (1, aRow), Max (1, fNumOfRows - aRow), aRect);
10 974 --          chunkFound := FindRowChunk (aRow, Num, theTotal, Index);
10 975 --
10 976 --          FOR i := 1 TO numOfRows DO
10 977 2-      BEGIN
10 978 --              fMaxVertical := fMaxVertical - fRowHeights^[Num].size;
10 979 --
10 980 --              fRowHeights^[Num].count := fRowHeights^[Num].count - 1;
10 981 --
10 982 --              IF (fRowHeights^[Num].count < Index) THEN
10 983 3-      BEGIN
10 984 --                  IF (fRowHeights^[Num].count = 0) THEN
10 985 4-      BEGIN
10 986 --                      (* need to delete that chunk *)
10 987 --                      offset := LONGINT (IntMultiply (4, Num));
10 988 --                      result := Munger (Handle (fRowHeights), offset, Ptr (Ord (fRowHeights^) + offset),
10 989 --                      sizeof (LONGINT), dummy, 0);

```

```

10 990 --          fNumOfRowChunks := fNumOfRowChunks - 1;
10 991 --
10 992 --          (* see if we can consolidate chunks *)
10 993 --          IF (Num > 0) & (fRowHeights^[Num - 1].size = fRowHeights^[Num].size) THEN
10 994 5-          BEGIN
10 995 --              fRowHeights^[Num - 1].count := fRowHeights^[Num - 1].count +
10 996 --                  fRowHeights^[Num].count;
10 997 --              (* need to delete that chunk *)
10 998 --              offset := LONGINT (IntMultiply (4, Num));
10 999 --              result := Munger (Handle (fRowHeights), offset, Ptr (Ord (fRowHeights) + offset),
10 1000 --                  sizeof (LONGINT), dummy, 0);
10 1001 --              fNumOfRowChunks := fNumOfRowChunks - 1;
10 1002 -5          END;
10 1003 -4          END
10 1004 --          ELSE
10 1005 4-          BEGIN
10 1006 --              Num := Num + 1;
10 1007 -4          END;
10 1008 --          Index := 1;
10 1009 -3          END;
10 1010 -2          END;
10 1011 --
10 1012 --          fNumOfRows := fNumOfRows - numOfRows;
10 1013 --
10 1014 --          (* reset run array info *)
10 1015 --          fLastRow := 0;
10 1016 --          fLastRowChunkNum := 0;
10 1017 --          fLastRowTotal := 0;
10 1018 --          fLastRowIndex := 1;
10 1019 --
10 1020 --          AdjustSize;
10 1021 --
10 1022 --          InvalidateVRect (aRect);
10 1023 -1          END;
10 1024 -0 A          END; (* TGridView.DelRowAt *)
10 1025 --
10 1026 --
10 1027 --          {*****}
10 1028 --          (* TGridView.DelColFirst *)
10 1029 --          (* Delete the first numOfCols. This method causes a re-calculation *)
10 1030 --          (* of the extent and redraws only the necessary cells (and adornments). *)
10 1031 --          {*****}
10 1032 --          ($S GVNonRes)
10 1033 -A          PROCEDURE TGridView.DelColFirst (numOfCols: INTEGER);
10 1034 0- A          BEGIN
10 1035 --              DelColAt (1, numOfCols);
10 1036 -0 A          END; (* TGridView.DelColFirst *)
10 1037 --
10 1038 --
10 1039 --          {*****}
10 1040 --          (* TGridView.DelRowFirst *)
10 1041 --          (* Delete the first numOfRows. This method causes a re-calculation *)
10 1042 --          (* of the extent and redraws only the necessary cells (and adornments). *)
10 1043 --          {*****}
10 1044 --          ($S GVNonRes)
10 1045 -A          PROCEDURE TGridView.DelRowFirst (numOfRows: INTEGER);
10 1046 0- A          BEGIN
10 1047 --              DelRowAt (1, numOfRows);
10 1048 -0 A          END; (* TGridView.DelRowFirst *)
10 1049 --
10 1050 --
10 1051 --          {*****}
10 1052 --          (* TGridView.DelColLast *)
10 1053 --          (* Delete the last numOfCols. This method causes a re-calculation *)
10 1054 --          (* of the extent and redraws only the necessary cells (and adornments). *)
10 1055 --          {*****}
10 1056 --          ($S GVNonRes)
10 1057 -A          PROCEDURE TGridView.DelColLast (numOfCols: INTEGER);
10 1058 0- A          BEGIN
10 1059 --              DelColAt (fNumOfCols - numOfCols + 1, numOfCols);
10 1060 -0 A          END; (* TGridView.DelColLast *)
10 1061 --
10 1062 --          {*****}
10 1063 --          (* TGridView.DelRowLast *)
10 1064 --          (* Delete the last numOfRows. This method causes a re-calculation *)
10 1065 --          (* of the extent and redraws only the necessary cells (and adornments). *)
10 1066 --          {*****}
10 1067 --          ($S GVNonRes)
10 1068 -A          PROCEDURE TGridView.DelRowLast (numOfRows: INTEGER);
10 1069 0- A          BEGIN
10 1070 --              DelRowAt (fNumOfRows - numOfRows + 1, numOfRows);
10 1071 -0 A          END; (* TGridView.DelRowLast *)
10 1072 --
10 1073 --
10 1074 --          {*****}
10 1075 --          (* TGridView.EachCellDo *)
10 1076 --          (* For each cell in the rectangle defined by startCell and stopCell, *)
10 1077 --          (* perform the procedure: DoToCell. startCell must be <= stopCell or the *)
10 1078 --          (* result is undefined. *)
10 1079 --          {*****}
10 1080 --          ($S GVRes)
10 1081 -A          PROCEDURE TGridView.EachCellDo (startCell, stopCell: GridCell;
10 1082 --              PROCEDURE DoToCell (aCell: GridCell));
10 1083 --          VAR
10 1084 --              i, j:          INTEGER;
10 1085 --              firstRow:      INTEGER;
10 1086 --              lastRow:        INTEGER;
10 1087 --              firstCol:        INTEGER;
10 1088 --              lastCol:         INTEGER;

```

```

10 1089 --      aCell:      GridCell;
10 1090 --
10 1091 0- A BEGIN
10 1092 --      firstRow := Max (1, startCell.v);
10 1093 --      firstCol := Max (1, startCell.h);
10 1094 --      lastRow  := Min (fNumOfRows, stopCell.v);
10 1095 --      lastCol  := Min (fNumOfCols, stopCell.h);
10 1096 --
10 1097 --      ($IFC qDebug)
10 1098 --      IF FALSE THEN
10 1099 1- BEGIN
10 1100 --          WRITELN ('TGridView.EachCellDo');
10 1101 --          WRITELN ('firstCol: ', firstCol:2, ' lastCol: ', lastCol:2);
10 1102 --          WRITELN ('firstRow: ', firstRow:2, ' lastRow: ', lastRow:2);
10 1103 --          WRITELN;
10 1104 -1 END;
10 1105 --      ($ENDC)
10 1106 --
10 1107 --
10 1108 --      FOR j := firstRow TO lastRow DO
10 1109 1- BEGIN
10 1110 --          FOR i := firstCol TO lastCol DO
10 1111 2- BEGIN
10 1112 --              aCell.v := j;
10 1113 --              aCell.h := i;
10 1114 --              DoToCell (aCell);
10 1115 -2 END;
10 1116 -1 END;
10 1117 -0 A END: (* TGridView.EachCellDo *)
10 1118 --
10 1119 --
10 1120 --      {*****}
10 1121 --      (* TGridView.EachSelectedCellDo *)
10 1122 --      (* For each cell in the selection region, perform the procedure: DoToCell. *)
10 1123 --      {*****}
10 1124 --      ($S GVRes)
10 1125 -- A PROCEDURE TGridView.EachSelectedCellDo (PROCEDURE DoToCell (aCell: GridCell));
10 1126 0- A BEGIN
10 1127 --      EachInRgn (fSelections, DoToCell);
10 1128 -0 A END: (* TGridView.EachSelectedCellDo *)
10 1129 --
10 1130 --
10 1131 --      {*****}
10 1132 --      (* TGridView.EachInRgn *)
10 1133 --      (* For each cell in the specified region perform the procedure DoToCell. *)
10 1134 --      (* This is the method to use to step through the region containing the *)
10 1135 --      (* current selections. *)
10 1136 --      {*****}
10 1137 --      ($S GVRes)
10 1138 -- A PROCEDURE TGridView.EachInRgn (aRgn: RgnHandle;
10 1139 --      PROCEDURE DoToCell (aCell: GridCell));
10 1140 --
10 1141 --      VAR
10 1142 --          boundRect:      Rect;
10 1143 --
10 1144 0- B PROCEDURE IfInRegion (aCell: GridCell);
10 1145 0- B BEGIN
10 1146 1- IF PtInRgn (aCell, aRgn) THEN
10 1147 --      DoToCell (aCell);
10 1148 -1 END;
10 1149 -0 B END;
10 1150 --
10 1151 0- A BEGIN
10 1152 --      boundRect := aRgn^.rgnBBox;
10 1153 --      EachCellDo (boundRect.TopLeft, boundRect.BotRight, IfInRegion);
10 1154 -0 A END: (* TGridView.EachInRgn *)
10 1155 --
10 1156 --
10 1157 --      {*****}
10 1158 --      (* TGridView.FindColChunk *)
10 1159 --      (* Chunks start at zero; items start at one; that is, the *)
10 1160 --      (* third width in the second chunk is in aChunkNum: 1 with Index: 3. *)
10 1161 --      {*****}
10 1162 --      ($S GVRes)
10 1163 -- A FUNCTION TGridView.FindColChunk (aCol: INTEGER; VAR aChunkNum: INTEGER;
10 1164 --      VAR theTotal: LONGINT; VAR indexInChunk: INTEGER): BOOLEAN;
10 1165 --
10 1166 --      VAR
10 1167 --          thisCol:      INTEGER;
10 1168 --          count:      INTEGER;
10 1169 --          deltaCols:  INTEGER;
10 1170 0- A BEGIN
10 1171 --      IF (fNumOfColChunks <= 0) | (aCol > fNumOfCols) | (aCol <= 0) THEN
10 1172 1- BEGIN
10 1173 --          aChunkNum := 0;
10 1174 --          theTotal := 0;
10 1175 --          indexInChunk := 0;
10 1176 --          FindColChunk := FALSE;
10 1177 --          aCol := 0;
10 1178 -1 END
10 1179 --      ELSE IF (aCol = fLastCol) THEN
10 1180 1- BEGIN (* check for the very easy case *)
10 1181 --          aChunkNum := fLastColChunkNum;
10 1182 --          theTotal := fLastColTotal;
10 1183 --          indexInChunk := fLastColIndex;
10 1184 --          FindColChunk := TRUE;
10 1185 -1 END
10 1186 --      ELSE
10 1187 1- BEGIN

```

```

10 1188 --      deltaCols := ABS (aCol - fLastCol);
10 1189 --
10 1190 --      IF (deltaCols >= aCol) | (aCol <= fColWidths^[0].count) THEN
10 1191 2- BEGIN (* start from the first chunk *)
10 1192 --          aChunkNum := 0;
10 1193 --          theTotal := 0;
10 1194 --          thisCol := 0;
10 1195 -2      END
10 1196 --      ELSE IF (deltaCols > (fNumOfCols - aCol + 1)) THEN
10 1197 2- BEGIN (* start from the end chunk *)
10 1198 --          aChunkNum := fNumOfColChunks - 1;
10 1199 --          count := fColWidths^[aChunkNum].count;
10 1200 --          theTotal := fMaxHorizontal -
10 1201 --              IntMultiply (count, fColWidths^[aChunkNum].size);
10 1202 --          thisCol := fNumOfCols - count;
10 1203 -2      END
10 1204 --      ELSE
10 1205 2- BEGIN (* start from the previous values *)
10 1206 --          aChunkNum := fLastColChunkNum;
10 1207 --          theTotal := fLastColTotal;
10 1208 --          thisCol := fLastCol - fLastColIndex;
10 1209 -2      END;
10 1210 --
10 1211 --      IF aCol > thisCol THEN
10 1212 2- BEGIN
10 1213 --          WHILE ((thisCol + fColWidths^[aChunkNum].count) < aCol) DO
10 1214 3- BEGIN
10 1215 --              count := fColWidths^[aChunkNum].count;
10 1216 --              theTotal := theTotal + IntMultiply (count, fColWidths^[aChunkNum].size);
10 1217 --              thisCol := thisCol + count;
10 1218 --              aChunkNum := aChunkNum + 1;
10 1219 -3          END;
10 1220 -2      END
10 1221 --      ELSE
10 1222 2- BEGIN
10 1223 3- REPEAT
10 1224 --          aChunkNum := aChunkNum - 1;
10 1225 --          count := fColWidths^[aChunkNum].count;
10 1226 --          theTotal := theTotal - IntMultiply (count, fColWidths^[aChunkNum].size);
10 1227 --          thisCol := thisCol - count;
10 1228 -3      UNTIL (thisCol <= aCol);
10 1229 -2      END;
10 1230 --          indexInChunk := aCol - thisCol;
10 1231 --          FindColChunk := TRUE;
10 1232 -1      END;
10 1233 --      (* cache the last values *)
10 1234 --      fLastCol := aCol;
10 1235 --      fLastColChunkNum := aChunkNum;
10 1236 --      fLastColTotal := theTotal;
10 1237 --      fLastColIndex := indexInChunk;
10 1238 -0 A END; (* TGridView.FindColChunk *)
10 1239 --
10 1240 --
10 1241 --      {*****}
10 1242 --      (* TGridView.FindRowChunk *)
10 1243 --      (* Chunks start at zero; Items start at one; that is, the *)
10 1244 --      (* third height in the second chunk is in aChunkNum: 1 with Index: 3. *)
10 1245 --      {*****}
10 1246 --      ($S GVRes)
10 1247 -- A FUNCTION TGridView.FindRowChunk (aRow: INTEGER; VAR aChunkNum: INTEGER;
10 1248 --      VAR theTotal: LONGINT; VAR indexInChunk: INTEGER): BOOLEAN;
10 1249 --      VAR
10 1250 --          thisRow: INTEGER;
10 1251 --          count: INTEGER;
10 1252 --          deltaRows: INTEGER;
10 1253 --
10 1254 0- A BEGIN
10 1255 --      IF (fNumOfRowChunks <= 0) | (aRow > fNumOfRows) | (aRow <= 0) THEN
10 1256 1- BEGIN
10 1257 --          aChunkNum := 0;
10 1258 --          theTotal := 0;
10 1259 --          indexInChunk := 0;
10 1260 --          FindRowChunk := FALSE;
10 1261 --          aRow := 0;
10 1262 -1      END
10 1263 --      ELSE IF (aRow = fLastRow) THEN
10 1264 1- BEGIN (* check for the very easy case *)
10 1265 --          aChunkNum := fLastRowChunkNum;
10 1266 --          theTotal := fLastRowTotal;
10 1267 --          indexInChunk := fLastRowIndex;
10 1268 --          FindRowChunk := TRUE;
10 1269 -1      END
10 1270 --      ELSE
10 1271 1- BEGIN
10 1272 --          deltaRows := ABS (aRow - fLastRow);
10 1273 --
10 1274 --          IF (deltaRows >= aRow) | (aRow <= fRowHeights^[0].count) THEN
10 1275 2- BEGIN (* start from the first chunk *)
10 1276 --              aChunkNum := 0;
10 1277 --              theTotal := 0;
10 1278 --              thisRow := 0;
10 1279 -2          END
10 1280 --          ELSE IF (deltaRows > (fNumOfRows - aRow + 1)) THEN
10 1281 2- BEGIN (* start from the end chunk *)
10 1282 --              aChunkNum := fNumOfRowChunks - 1;
10 1283 --              count := fRowHeights^[aChunkNum].count;
10 1284 --              theTotal := fMaxVertical -
10 1285 --                  IntMultiply (count, fRowHeights^[aChunkNum].size);
10 1286 --              thisRow := fNumOfRows - count;

```

```

10 1287 --      END
10 1288 --      ELSE
10 1289 --      BEGIN (* start from the previous values *)
10 1290 --          aChunkNum := fLastRowChunkNum;
10 1291 --          theTotal := fLastRowTotal;
10 1292 --          thisRow := fLastRow - fLastRowIndex;
10 1293 --      END;
10 1294 --
10 1295 --      IF aRow > thisRow THEN
10 1296 --      BEGIN
10 1297 --          WHILE ((thisRow + fRowHeights^[aChunkNum].count) < aRow) DO
10 1298 --          BEGIN
10 1299 --              count := fRowHeights^[aChunkNum].count;
10 1300 --              theTotal := theTotal + IntMultiply (count, fRowHeights^[aChunkNum].size);
10 1301 --              thisRow := thisRow + count;
10 1302 --              aChunkNum := aChunkNum + 1;
10 1303 --          END;
10 1304 --      END
10 1305 --      ELSE
10 1306 --      BEGIN
10 1307 --          REPEAT
10 1308 --              aChunkNum := aChunkNum - 1;
10 1309 --              count := fRowHeights^[aChunkNum].count;
10 1310 --              theTotal := theTotal - IntMultiply (count, fRowHeights^[aChunkNum].size);
10 1311 --              thisRow := thisRow - count;
10 1312 --          UNTIL (thisRow <= aRow);
10 1313 --      END;
10 1314 --      indexInChunk := aRow - thisRow;
10 1315 --      FindRowChunk := TRUE;
10 1316 --      END;
10 1317 --      (* cache the last values *)
10 1318 --      fLastRow := aRow;
10 1319 --      fLastRowChunkNum := aChunkNum;
10 1320 --      fLastRowTotal := theTotal;
10 1321 --      fLastRowIndex := indexInChunk;
10 1322 -- A END; (* TGridView.FindRowChunk *)
10 1323 --
10 1324 --
10 1325 --      {*****}
10 1326 --      (* TGridView.FirstSelectedCell *)
10 1327 --      (* Returns the top left cell in the selected range, if any *)
10 1328 --      {*****}
10 1329 --      ($$ GVRes)
10 1330 -- A FUNCTION TGridView.FirstSelectedCell: GridCell;
10 1331 --
10 1332 -- A BEGIN
10 1333 --      FirstSelectedCell := fSelections^.rgnBBBox.topLeft;
10 1334 -- A END; (* TGridView.FirstSelectedCell *)
10 1335 --
10 1336 --
10 1337 --      {*****}
10 1338 --      (* TGridView.GetColWidth *)
10 1339 --      (* Get the column width for the given column. This includes insets. *)
10 1340 --      {*****}
10 1341 --      ($$ GVRes)
10 1342 -- A FUNCTION TGridView.GetColWidth (aCol: INTEGER): INTEGER;
10 1343 --      VAR
10 1344 --          Num: INTEGER;
10 1345 --          theTotal: LONGINT;
10 1346 --          Index: INTEGER;
10 1347 --
10 1348 -- A BEGIN
10 1349 --      ($IFC qRangeCheck)
10 1350 --          IF (aCol < 1) OR (aCol > fNumOfCols) THEN
10 1351 --          BEGIN
10 1352 --              ($IFC qDebug)
10 1353 --                  Writeln('fNumOfCols = ', fNumOfCols:1, ' aCol = ', aCol:1);
10 1354 --              ($ENDC)
10 1355 --              ProgramBreak('Range Check in GetColWidth');
10 1356 --          END;
10 1357 --      ($ENDC)
10 1358 --
10 1359 --          IF fNumOfColChunks = 1 THEN
10 1360 --              GetColWidth := fColWidths^[0].size
10 1361 --          ELSE IF FindColChunk (aCol, Num, theTotal, Index) THEN
10 1362 --              GetColWidth := fColWidths^[Num].size
10 1363 --          ELSE
10 1364 --              GetColWidth := 0;
10 1365 -- A END; (* TGridView.GetColWidth *)
10 1366 --
10 1367 --
10 1368 --      {*****}
10 1369 --      (* TGridView.GetRowHeight *)
10 1370 --      (* Get the row height for the given row. This includes insets. *)
10 1371 --      {*****}
10 1372 --      ($$ GVRes)
10 1373 -- A FUNCTION TGridView.GetRowHeight (aRow: INTEGER): INTEGER;
10 1374 --      VAR
10 1375 --          Num: INTEGER;
10 1376 --          theTotal: LONGINT;
10 1377 --          Index: INTEGER;
10 1378 --
10 1379 -- A BEGIN
10 1380 --      ($IFC qRangeCheck)
10 1381 --          IF (aRow < 1) | (aRow > fNumOfRows) THEN
10 1382 --          BEGIN
10 1383 --              ($IFC qDebug)
10 1384 --                  Writeln('fNumOfRows = ', fNumOfRows:1, ' aRow = ', aRow:1);
10 1385 --              ($ENDC)

```

```

10 1386 --      ProgramBreak('Range Check in GetRowHeight');
10 1387 -1      END;
10 1388 --      ($ENDC)
10 1389 --
10 1390 --      IF fNumOfRowChunks = 1 THEN
10 1391 --          GetRowHeight := fRowHeights^[0].size
10 1392 --      ELSE IF FindRowChunk (aRow, Num, theTotal, Index) THEN
10 1393 --          GetRowHeight := fRowHeights^[Num].size
10 1394 --      ELSE
10 1395 --          GetRowHeight := 0;
10 1396 -0 A  END; (* TGridView.GetRowHeight *)
10 1397 --
10 1398 --
10 1399 --      {*****}
10 1400 --      (* TGridView.InvalidateCell *)
10 1401 --      (* Mark a cell invalid. This forces a re-draw of the specified cell. *)
10 1402 --      {*****}
10 1403 --      ($S GVRes)
10 1404 -0 A  PROCEDURE TGridView.InvalidateCell (aCell: GridCell);
10 1405 --      VAR
10 1406 --          aRect: VRect;
10 1407 --
10 1408 -0 A  BEGIN
10 1409 --      CellToVRect (aCell, aRect);
10 1410 --      InvalidVRect (aRect);
10 1411 -0 A  END; (* TGridView.InvalidateCell *)
10 1412 --
10 1413 --
10 1414 --      {*****}
10 1415 --      (* TGridView.InvalidateSelection *)
10 1416 --      (* Mark the rectangle bounding all the selections invalid. *)
10 1417 --      (* This forces a re-draw of the enclosed cells. *)
10 1418 --      {*****}
10 1419 --      ($S GVRes)
10 1420 -0 A  PROCEDURE TGridView.InvalidateSelection;
10 1421 --      VAR
10 1422 --          boundRect: VRect;
10 1423 --          aRect: VRect;
10 1424 --          BBox: Rect;
10 1425 --
10 1426 -0 A  BEGIN
10 1427 --      BBox := fSelections^.rgnBBox;
10 1428 --
10 1429 --      BBox.bottom := BBox.bottom - 1;
10 1430 --      BBox.right := BBox.right - 1;
10 1431 --      CellToVRect (BBox.topleft, boundRect);
10 1432 --      CellToVRect (BBox.botright, aRect);
10 1433 --
10 1434 --      boundRect.bottom := aRect.bottom;
10 1435 --      boundRect.right := aRect.right;
10 1436 --
10 1437 --      InvalidVRect (boundRect);
10 1438 -0 A  END; (* TGridView.InvalidateSelection *)
10 1439 --
10 1440 --
10 1441 --      {*****}
10 1442 --      (* TGridView.IdentifyPoint *)
10 1443 --      (* Given a point in view coordinates, return the row and column and the *)
10 1444 --      (* part of the view in which the point lies (inCell, inRow, inColumn, *)
10 1445 --      (* inVertex). *)
10 1446 --      {*****}
10 1447 --      ($S GVRes)
10 1448 -0 A  FUNCTION TGridView.IdentifyPoint (theQDPoint: Point;
10 1449 --      VAR aRow, aCol: INTEGER): GridViewPart;
10 1450 --      VAR
10 1451 --          aRect: VRect;
10 1452 --          aPoint: VPoint;
10 1453 --          aCell: GridCell;
10 1454 --
10 1455 -0 A  BEGIN
10 1456 --      QDToViewPt (theQDPoint, aPoint);
10 1457 --      aCell := VPointToCell (aPoint);
10 1458 --      CellToVRect (aCell, aRect);
10 1459 --      InsetVRect (aRect, fColInset DIV 2, fRowInset DIV 2);
10 1460 --      aRow := aCell.v;
10 1461 --      aCol := aCell.h;
10 1462 --
10 1463 --      IF (aCell.v > fMaxVertical) | (aCell.h > fMaxHorizontal) |
10 1464 --      (aCell.v < 0) | (aCell.h < 0) THEN
10 1465 -1-      BEGIN
10 1466 --          IdentifyPoint := badChoice;
10 1467 -1-      END
10 1468 --      ELSE
10 1469 -1-      BEGIN
10 1470 --          IdentifyPoint := inCell;
10 1471 --
10 1472 --          IF fColInset > 0 THEN
10 1473 -2-      BEGIN
10 1474 --          IF (aPoint.h < aRect.left) THEN
10 1475 -3-      BEGIN
10 1476 --          IdentifyPoint := InColumn; { To the left of the cell }
10 1477 -3-      END
10 1478 --
10 1479 --          ELSE IF (aPoint.h > aRect.right) THEN
10 1480 -3-      BEGIN
10 1481 --          IdentifyPoint := InColumn; { To the right of the cell }
10 1482 --          aCol := aCol + 1;
10 1483 -3-      END;
10 1484 -2-      END;

```

```

10 1485 --
10 1486 --           IF fRowInset > 0 THEN
10 1487 2-           BEGIN
10 1488 --               IF (aPoint.v < aRect.top) THEN
10 1489 3-               BEGIN
10 1490 --                   IF (IdentifyPoint = InColumn) THEN
10 1491 --                       IdentifyPoint := InVertex ( Click on both )
10 1492 --                   ELSE
10 1493 --                       IdentifyPoint := InRow; ( Above the cell )
10 1494 -3                   END
10 1495 --               ELSE IF (aPoint.v > aRect.bottom) THEN
10 1496 --               BEGIN
10 1497 3-                   IF (IdentifyPoint = InColumn) THEN
10 1498 --                       IdentifyPoint := InVertex ( Click on both )
10 1499 --                   ELSE
10 1500 --                       IdentifyPoint := InRow; ( Above the cell )
10 1501 --                       aRow := aRow + 1;
10 1502 --                   END;
10 1503 -3               END;
10 1504 -2           END;
10 1505 -1       END;
10 1506 -0 A   END; (* TGridView.IdentifyPoint *)
10 1507 --
10 1508 --
10 1509 -- {*****}
10 1510 -- { * TGridView.InsColBefore * }
10 1511 -- { *   Insert numOfCols starting at aCol. This method causes a re-calculation * }
10 1512 -- { *   of the extent and redraws only the necessary cells (and adornments). * }
10 1513 -- {*****}
10 1514 -- { $$ GVRes }
10 1515 -- A   PROCEDURE TGridView.InsColBefore (aCol :INTEGER; numOfCols: INTEGER; aWidth: INTEGER);
10 1516 --   VAR
10 1517 --       oldSize:      Size;
10 1518 --       offset:       LONGINT;
10 1519 --       result:       LONGINT;
10 1520 --       aNumber:      LONGINT;
10 1521 --       aRect:        VRect;
10 1522 --       Num:          INTEGER;
10 1523 --       theTotal:     LONGINT;
10 1524 --       Index:        INTEGER;
10 1525 --
10 1526 0- A   BEGIN
10 1527 --       IF (aCol < 1) | (numOfCols < 1) THEN
10 1528 1-       BEGIN
10 1529 --           IF numOfCols <> 0 THEN
10 1530 2-       BEGIN
10 1531 --           ($IFC qRangeCheck)
10 1532 --           ($IFC qDebug)
10 1533 --               Writeln('fNumOfCols = ', fNumOfCols:1, ' aCol = ', aCol:1);
10 1534 --           ($ENDC)
10 1535 --               ProgramBreak('Range Check in InsColBefore');
10 1536 --           ($ENDC)
10 1537 -2       END;
10 1538 -1       END
10 1539 --       ELSE
10 1540 1-       BEGIN
10 1541 --           { * Check if we can just increment the last size count * }
10 1542 --           IF (aCol > fNumOfCols) & (fNumOfColChunks > 0) &
10 1543 --              (fColWidths^[fNumOfColChunks-1].size = aWidth) THEN
10 1544 2-       BEGIN { * Just need to increment the ctr * }
10 1545 --           fColWidths^[fNumOfColChunks-1].count :=
10 1546 --           fColWidths^[fNumOfColChunks-1].count + numOfCols;
10 1547 -2       END
10 1548 --           { * check if we can increment any size count * }
10 1549 --           ELSE IF FindColChunk (aCol, Num, theTotal, Index) &
10 1550 --              (fColWidths^[Num].size = aWidth) THEN
10 1551 --           BEGIN
10 1552 2-       BEGIN
10 1553 --           fColWidths^[Num].count := fColWidths^[Num].count + numOfCols;
10 1554 -2       END
10 1555 --           { * We need to create a new chunk, possibly two * }
10 1556 --           ELSE
10 1557 --           BEGIN
10 1558 2-       BEGIN
10 1559 --           oldSize := GetHandleSize (Handle (fColWidths));
10 1560 --           aNumber := 0;
10 1561 --           IF (Index <= 1) | (aCol > fNumOfCols) THEN
10 1562 --           BEGIN { * need to add one chunk * }
10 1563 3-       BEGIN
10 1564 --           IF (aCol > fNumOfCols) THEN
10 1565 4-       BEGIN { * add a row on the end * }
10 1566 --           Num := fNumOfColChunks;
10 1567 -4       END;
10 1568 --           offset := LONGINT (IntMultiply (Num, 4));
10 1569 --           result := Munger (Handle (fColWidths), offset, NIL, 0, Ptr(@aNumber),
10 1570 --                           Sizeof (LONGINT));
10 1571 --           IF GetHandleSize (Handle (fColWidths)) <= oldSize THEN
10 1572 --           Failure (memFullErr, 0);
10 1573 --           fColWidths^[Num].size := aWidth;
10 1574 --           fColWidths^[Num].count := numOfCols;
10 1575 --           fNumOfColChunks := fNumOfColChunks + 1;
10 1576 --           END
10 1577 --           ELSE
10 1578 --           BEGIN
10 1579 -3       BEGIN
10 1580 --           { * need to add two * }
10 1581 3-       BEGIN
10 1582 --           offset := LONGINT (IntMultiply (Num+1, 4));
10 1583 --

```





```

10 1683 --      offset := LONGINT (IntMultiply (Num+1, 4));
10 1684 --      result := Munger (Handle (fRowHeights) , offset, NIL, 0, Ptr(@aNumber),
10 1685 --                    2 * sizeof (LONGINT));
10 1686 --
10 1687 --      IF GetHandleSize (Handle (fRowHeights)) <= oldSize THEN
10 1688 --          Failure (memFullErr, 0);
10 1689 --
10 1690 --          fRowHeights^[Num+2].count := fRowHeights^[Num].count - Index + 1;
10 1691 --          fRowHeights^[Num+2].size := fRowHeights^[Num].size;
10 1692 --          fRowHeights^[Num].count := Index - 1;
10 1693 --          fRowHeights^[Num+1].size := aHeight;
10 1694 --          fRowHeights^[Num+1].count := numOfRows;
10 1695 --          fNumOfRowChunks := fNumOfRowChunks + 2;
10 1696 -3      END;
10 1697 -2      END;
10 1698 --
10 1699 --      fNumOfRows := fNumOfRows + numOfRows;
10 1700 --      fMaxVertical := fMaxVertical + IntMultiply (numOfRows, aHeight);
10 1701 --      AdjustSize;
10 1702 --
10 1703 --      RowToVRect (Max (1, aRow), Max (1, fNumOfRows - aRow + 1), aRect);
10 1704 --      InvalidVRect (aRect);
10 1705 -1      END;
10 1706 -0 A  END; (* TGridView.InsRowBefore *)
10 1707 --
10 1708 --
10 1709 --      {*****}
10 1710 --      (* TGridView.InsColLast *)
10 1711 --      (* Insert numOfCols at the end. This method causes a re-calculation *)
10 1712 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 1713 --      {*****}
10 1714 --      ($$ GVRes)
10 1715 -- A  PROCEDURE TGridView.InsColLast (numOfCols: INTEGER; aWidth: INTEGER);
10 1716 0- A  BEGIN
10 1717 --      InsColBefore (fNumOfCols + 1, numOfCols, aWidth);
10 1718 -0 A  END; (* TGridView.InsColLast *)
10 1719 --
10 1720 --
10 1721 --      {*****}
10 1722 --      (* TGridView.InsRowLast *)
10 1723 --      (* Insert numOfRows at the end. This method causes a re-calculation *)
10 1724 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 1725 --      {*****}
10 1726 --      ($$ GVRes)
10 1727 -- A  PROCEDURE TGridView.InsRowLast (numOfRows: INTEGER; aHeight: INTEGER);
10 1728 0- A  BEGIN
10 1729 --      InsRowBefore (fNumOfRows + 1, numOfRows, aHeight);
10 1730 -0 A  END; (* TGridView.InsRowLast *)
10 1731 --
10 1732 --
10 1733 --      {*****}
10 1734 --      (* TGridView.InsColFirst *)
10 1735 --      (* Insert numOfCols at the start. This method causes a re-calculation *)
10 1736 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 1737 --      {*****}
10 1738 --      ($$ GVRes)
10 1739 -- A  PROCEDURE TGridView.InsColFirst (numOfCols: INTEGER; aWidth: INTEGER);
10 1740 0- A  BEGIN
10 1741 --      InsColBefore (kOneCol, numOfCols, aWidth);
10 1742 -0 A  END; (* TGridView.InsColFirst *)
10 1743 --
10 1744 --
10 1745 --      {*****}
10 1746 --      (* TGridView.InsRowFirst *)
10 1747 --      (* Insert numOfRows at the start. This method causes a re-calculation *)
10 1748 --      (* of the extent and redraws only the necessary cells (and adornments). *)
10 1749 --      {*****}
10 1750 --      ($$ GVRes)
10 1751 -- A  PROCEDURE TGridView.InsRowFirst (numOfRows: INTEGER; aHeight: INTEGER);
10 1752 0- A  BEGIN
10 1753 --      InsRowBefore (kOneRow, numOfRows, aHeight);
10 1754 -0 A  END; (* TGridView.InsRowFirst *)
10 1755 --
10 1756 --
10 1757 --      {*****}
10 1758 --      (* TGridView.IsCellSelected *)
10 1759 --      (* Check if the specified cell is currently selected. *)
10 1760 --      {*****}
10 1761 --      ($$ GVRes)
10 1762 -- A  FUNCTION TGridView.IsCellSelected (aCell: GridCell) : BOOLEAN;
10 1763 0- A  BEGIN
10 1764 --      IsCellSelected := PtInRgn (aCell, fSelections);
10 1765 -0 A  END; (* TGridView.IsCellSelected *)
10 1766 --
10 1767 --
10 1768 --      {*****}
10 1769 --      (* TGridView.LastSelectedCell *)
10 1770 --      (* Returns the bottom right cell in the selected range, if any *)
10 1771 --      {*****}
10 1772 --      ($$ GVRes)
10 1773 -- A  FUNCTION TGridView.LastSelectedCell: GridCell;
10 1774 0- A  BEGIN
10 1775 --      LastSelectedCell := fSelections^.rgnBBot.botRight;
10 1776 -0 A  END; (* TGridView.LastSelectedCell *)
10 1777 --
10 1778 --
10 1779 --      {*****}
10 1780 --      (* TGridView.RowToVRect *)
10 1781 --      (* Calculate the bounding rectangle for the given row(s). The left and

```

```

10 1782 --      (*      right sides of the rectangle are the same as the view.      *)
10 1783 --      (*      The rectangle includes the cells in that row and the row inset.      *)
10 1784 --      {*****}
10 1785 --      {$S GVRes}
10 1786 -- A  PROCEDURE TGridView.RowToVRect (aRow: INTEGER; numOfRows: INTEGER; VAR aRect: VRect);
10 1787 --      VAR
10 1788 --          height:    LONGINT;
10 1789 --          Num:        INTEGER;
10 1790 --          theTotal:    LONGINT;
10 1791 --          Index:       INTEGER;
10 1792 --          i:           INTEGER;
10 1793 --
10 1794 0- A  BEGIN
10 1795 --      IF (aRow < 1) | (numOfRows < 1) | (aRow + numOfRows - 1 > fNumOfRows) THEN
10 1796 1-      BEGIN
10 1797 --          {$IFC qRangeCheck}
10 1798 --          {$IFC qDebug}
10 1799 --              Writeln('fNumOfRows = ', fNumOfRows:1, ' aRow = ', aRow:1);
10 1800 --          {$ENDC}
10 1801 --          ProgramBreak('Range Check in RowToVRect');
10 1802 --          {$ENDC}
10 1803 --          SetVRect (aRect, 0, 0, 0, 0);
10 1804 -1      END
10 1805 --      ELSE      (* all the params look OK *)
10 1806 1-      BEGIN
10 1807 --          height := 0;
10 1808 --
10 1809 --          IF (fNumOfRowChunks = 1) THEN      (* only one row height *)
10 1810 2-      BEGIN
10 1811 --              num := 0;
10 1812 --              theTotal := IntMultiply (fRowHeights^[num].size, aRow-1);
10 1813 --              height := IntMultiply (numOfRows, fRowHeights^[num].size);
10 1814 -2      END
10 1815 --          ELSE
10 1816 2-      BEGIN
10 1817 --              IF FindRowChunk (aRow, Num, theTotal, Index) THEN
10 1818 3-      BEGIN
10 1819 --                  theTotal := theTotal + IntMultiply ((Index-1), fRowHeights^[Num].size);
10 1820 --                  For i := aRow TO (aRow + numOfRows - 1) DO
10 1821 --                      height := height + GetRowHeight (i);
10 1822 -3      END;
10 1823 -2      END;
10 1824 --
10 1825 --          WITH aRect DO
10 1826 2-      BEGIN
10 1827 --              top      := theTotal;
10 1828 --              left     := 0;
10 1829 --              bottom   := theTotal + height;
10 1830 --              right    := fMaxHorizontal;
10 1831 -2      END;
10 1832 -1      END;
10 1833 0- A  END; (* TGridView.RowToVRect *)
10 1834 --
10 1835 --      {*****}
10 1836 --      (* TGridView.SameCell *)
10 1837 --      (*      Check if the cells are the same.      *)
10 1838 --      {*****}
10 1839 --      {$S GVRes}
10 1840 --
10 1841 -- A  FUNCTION TGridView.SameCell (aCell, bCell: GridCell): BOOLEAN;
10 1842 0- A  BEGIN
10 1843 --      SameCell := EqualPt (aCell, bCell);
10 1844 0- A  END; (* TGridView.SameCell *)
10 1845 --
10 1846 --      {*****}
10 1847 --      (* TGridView.ScrollSelectionIntoView *)
10 1848 --      (*      Scroll the selection (or a piece of it) into view.      *)
10 1849 --      {*****}
10 1850 --      {$S GVRes}
10 1851 --
10 1852 -- A  PROCEDURE TGridView.ScrollSelectionIntoView (redraw: BOOLEAN);
10 1853 --      VAR
10 1854 --          botRightCell:    GridCell;
10 1855 --          topLeftRect,
10 1856 --          botRightRect,
10 1857 --          selectionRect:    VRect;
10 1858 --          minToSee:         Point;
10 1859 0- A  BEGIN
10 1860 --      IF NOT (EmptyRgn (fSelections)) THEN
10 1861 1-      BEGIN
10 1862 --          CellToVRect(fSelections^.rgnBBox.topLeft, topLeftRect);
10 1863 --          SetPt(botRightCell, fSelections^.rgnBBox.right - 1,
10 1864 --              fSelections^.rgnBBox.bottom - 1);
10 1865 --          CellToVRect(botRightCell, botRightRect);
10 1866 --          UnionVRect(topLeftRect, botRightRect, selectionRect);
10 1867 --          minToSee.v := Max(topLeftRect.bottom - topLeftRect.top,
10 1868 --              botRightRect.bottom - botRightRect.top);
10 1869 --          minToSee.h := Max(topLeftRect.right - topLeftRect.left,
10 1870 --              botRightRect.right - botRightRect.left);
10 1871 --          RevealRect (selectionRect, minToSee, redraw);
10 1872 -1      END;
10 1873 0- A  END; (* TGridView.ScrollSelectionIntoView *)
10 1874 --
10 1875 --      {*****}
10 1876 --      (* TGridView.SetColWidth *)
10 1877 --      (*      Set the width of the specified column to aWidth. This includes *)
10 1878 --      (*      column insets. *)
10 1879 --      {*****}
10 1880 --

```

```

10 1881 --      {$S GVNonRes}
10 1882 -- A  PROCEDURE TGridView.SetColWidth (aCol: INTEGER; numOfCols: INTEGER; aWidth: INTEGER);
10 1883 -- A  BEGIN
10 1884 --      {$IFC qRangeCheck}
10 1885 --          IF (aCol < 1) | (numOfCols < 1) | (aCol + numOfCols - 1 > fNumOfCols) THEN
10 1886 --              BEGIN
10 1887 --                  {$IFC qDebug}
10 1888 --                      Writeln('fNumOfCols = ', fNumOfCols:1, ' aCol = ', aCol:1);
10 1889 --                  ($ENDC)
10 1890 --                      ProgramBreak('Range Check in SetColWidth');
10 1891 --                  END;
10 1892 --              ($ENDC)
10 1893 --          IF NOT ((fNumOfColChunks = 1) & (fColWidths^[0].size = aWidth)) THEN
10 1894 --              BEGIN
10 1895 --                  DelColAt (aCol, numOfCols);
10 1896 --                  InsColBefore (aCol, numOfCols, aWidth);
10 1897 --              END;
10 1898 --          END;
10 1899 -- A  END; (* TGridView.SetColWidth *)
10 1900 --
10 1901 --
10 1902 --      {*****}
10 1903 --      (* TGridView.SetRowHeight *)
10 1904 --      (* Set the height of the specified row to Height. This includes *)
10 1905 --      (* column insets. *)
10 1906 --      {*****}
10 1907 --      {$S GVNonRes}
10 1908 -- A  PROCEDURE TGridView.SetRowHeight (aRow: INTEGER; numOfRows: INTEGER; aHeight: INTEGER);
10 1909 -- A  BEGIN
10 1910 --      {$IFC qRangeCheck}
10 1911 --          IF (aRow < 1) | (numOfRows < 1) | (aRow + numOfRows - 1 > fNumOfRows) THEN
10 1912 --              BEGIN
10 1913 --                  {$IFC qDebug}
10 1914 --                      Writeln('fNumOfRows = ', fNumOfRows:1, ' aRow = ', aRow:1);
10 1915 --                  ($ENDC)
10 1916 --                      ProgramBreak('Range Check in SetRowHeight');
10 1917 --                  END;
10 1918 --              ($ENDC)
10 1919 --          IF NOT ((fNumOfRowChunks = 1) & (fRowHeights^[0].size = aHeight)) THEN
10 1920 --              BEGIN
10 1921 --                  DelRowAt (aRow, numOfRows);
10 1922 --                  InsRowBefore (aRow, numOfRows, aHeight);
10 1923 --              END;
10 1924 --          END;
10 1925 -- A  END; (* TGridView.SetRowHeight *)
10 1926 --
10 1927 --
10 1928 --      {*****}
10 1929 --      (* TGridView.SelectCell *)
10 1930 --      (* Set the current selection to the specified cell and call SetSelection. *)
10 1931 --      {*****}
10 1932 --      {$S GVRes}
10 1933 -- A  PROCEDURE TGridView.SelectCell (theCell: GridCell; extend, highlight, select: BOOLEAN);
10 1934 -- A  BEGIN
10 1935 --      (* setup the region and then call SetSelection *)
10 1936 --      IF (theCell.h = 0) | (theCell.v = 0) THEN
10 1937 --          SetEmptyRgn (fTmpSelRgn)
10 1938 --      ELSE
10 1939 --          SetRectRgn (fTmpSelRgn, theCell.h, theCell.v, theCell.h + 1, theCell.v + 1);
10 1940 --      fWholeRect := TRUE;
10 1941 --      SetSelection (fTmpSelRgn, extend, highlight, select);
10 1942 -- A  END; (* TGridView.SelectCell *)
10 1943 --
10 1944 --
10 1945 --      {*****}
10 1946 --      (* TGridView.SetEmptySelection *)
10 1947 --      (* Set the current selection to be empty (nothing). "highlight" causes *)
10 1948 --      (* re-draw. *)
10 1949 --      {*****}
10 1950 --      {$S GVRes}
10 1951 -- A  PROCEDURE TGridView.SetEmptySelection (highlight: BOOLEAN);
10 1952 -- A  BEGIN
10 1953 --      SetEmptyRgn (fTmpSelRgn);
10 1954 --      SetSelection (fTmpSelRgn, kDontExtend, highlight, kSelect);
10 1955 --      fWholeRect := TRUE;
10 1956 -- A  END; (* TGridView.SetEmptySelection *)
10 1957 --
10 1958 --
10 1959 --      {*****}
10 1960 --      (* TGridView.SetSelection *)
10 1961 --      (* Set the current selections to the specified region. If extend is *)
10 1962 --      (* kExtend, theRegion is "added" to that already selected. If highlight *)
10 1963 --      (* is kHighlight, theRegion is highlighted ("highlight" causes an *)
10 1964 --      (* update to the screen. *)
10 1965 --      (* *)
10 1966 --      (* *)
10 1967 --      (* The field fWholeRect is used to determine if we need to do hilighting *)
10 1968 --      (* cell by cell, or for a whole rectangle of cells. If you call this *)
10 1969 --      (* method explicitly, then be sure to set fWholeRect appropriately before *)
10 1970 --      (* calling SetSelection. fLastWholeRect is used to determine if un- *)
10 1971 --      (* hilighting can be done by rectangle. It is used and set by this method *)
10 1972 --      (* only! fWholeRect = FALSE will work properly for ALL cases, but *)
10 1973 --      (* will be unnecessarily slow for the rectangle cases! *)
10 1974 --      (* *)
10 1975 --      (* The select parameter is used to select or de-select theRegion. If *)
10 1976 --      (* select is kDeSelect then extend is ignored and highlight is used to *)
10 1977 --      (* determine if an update is necessary. *)
10 1978 --      (* *)
10 1979 --      (* I assume here that if SetSelection is called explicitly (not by the *)

```

```

10 1980 --      (*      mouse-tracking code) then you know what you're doing in regards to      *)
10 1981 --      (*      single selections i.e. there is no checking to enforce single selection!*)
10 1982 --      {*****}
10 1983 --      {$S GVRes}
10 1984 -- A  PROCEDURE TGridView.SetSelection (theRegion: RgnHandle;
10 1985 --                                     extend, highlight, select: BOOLEAN);
10 1986 --      VAR
10 1987 --          hlRect:      Rect;
10 1988 --          hloffRect:  Rect;
10 1989 --          hlDesired:  HLState;
10 1990 --
10 1991 0- A  BEGIN
10 1992 --      IF highlight THEN
10 1993 1-  BEGIN
10 1994 --          (* get the ones to turn on *)
10 1995 --          CreateHighlightRgn (theRegion, fHLRegion, fWholeRect);
10 1996 --
10 1997 --          (* figure out what we need to turn off *)
10 1998 --          CreateHighlightRgn (fSelections, fTmpRgn, fLastWholeRect);
10 1999 --          CopyRgn (fTmpRgn, fTmpHLRgn);
10 2000 --
10 2001 --          IF fHLDesired = hLDim THEN
10 2002 --              hlDesired := hLOff;
10 2003 --
10 2004 --          IF (NOT extend & select) THEN      (* we need to turn off the old ones *)
10 2005 2-  BEGIN
10 2006 --              DiffRgn (fTmpRgn, fHLRegion, fTmpRgn);
10 2007 --              DoHilite (fTmpRgn, fHLDesired, hLOff);
10 2008 -2  END;
10 2009 --
10 2010 --          IF select THEN
10 2011 2-  BEGIN
10 2012 --              DiffRgn (fHLRegion, fTmpHLRgn, fHLRegion);
10 2013 --              DoHilite (fHLRegion, hLOff, fHLDesired);
10 2014 -2  END
10 2015 --          ELSE
10 2016 2-  BEGIN
10 2017 --              SectRgn (fHLRegion, fTmpHLRgn, fHLRegion);
10 2018 --              DoHilite (fHLRegion, fHLDesired, hLOff);
10 2019 -2  END;
10 2020 -1  END;
10 2021 --
10 2022 --          IF extend & select THEN (* extend the selection region *)
10 2023 --              UnionRgn (theRegion, fSelections, fSelections)
10 2024 --          ELSE IF select THEN      (* reset the selection region *)
10 2025 --              CopyRgn (theRegion, fSelections)
10 2026 --          ELSE                      (* need to de-select the new region *)
10 2027 1-  BEGIN
10 2028 --              fWholeRect := FALSE;
10 2029 --              DiffRgn (fSelections, theRegion, fSelections);
10 2030 -1  END;
10 2031 --
10 2032 --          CopyRgn (fSelections, fHLRegion);
10 2033 --          fLastWholeRect := fWholeRect;
10 2034 0- A  END; (* TGridView.SetSelection *)
10 2035 --
10 2036 --
10 2037 --      {*****}
10 2038 --      (* TGridView.SetSelRect *)
10 2039 --      (*      Set the current selections to the specified rectangle. If extend *)
10 2040 --      (*      is kExtend then the rectangle is "added" to that already selected, *)
10 2041 --      (*      if highlight is kHighlight then the rectangle is highlighted as well. *)
10 2042 --      {*****}
10 2043 --      {$S GVRes}
10 2044 -- A  PROCEDURE TGridView.SetSelRect (theTop, theLeft, theBottom, theRight: INTEGER;
10 2045 --                                  extend, highlight, select: BOOLEAN);
10 2046 0- A  BEGIN
10 2047 --      SetRectRgn (fTmpSelRgn, theLeft, theTop, theRight, theBottom);
10 2048 --      fWholeRect := TRUE; (* we know the selection is a rectangle *)
10 2049 --      SetSelection (fTmpSelRgn, extend, highlight, select);
10 2050 0- A  END; (* TGridView.SetSelRect *)
10 2051 --
10 2052 --
10 2053 --      {*****}
10 2054 --      (* TGridView.SetSingleSelection *)
10 2055 --      (*      If TRUE then only one item/cell can be selected at a time. *)
10 2056 --      {*****}
10 2057 --      {$S GVNonRes}
10 2058 -- A  PROCEDURE TGridView.SetSingleSelection (theSetting: BOOLEAN);
10 2059 0- A  BEGIN
10 2060 --      fSingleSelection := theSetting;
10 2061 --      fWholeRect := TRUE; (* we know the selection is a rectangle *)
10 2062 0- A  END; (* TGridView.SetSingleSelection *)
10 2063 --
10 2064 --
10 2065 --      {*****}
10 2066 --      (* TGridView.VPointToCell *)
10 2067 --      (*      Get the Cell in which the specified point lies. Points in the insets *)
10 2068 --      (*      of cells will be considered as inside that cell. *)
10 2069 --      (*      This method returns the cell (0,0) if the point is not within a cell. *)
10 2070 --      {*****}
10 2071 --      {$S GVRes}
10 2072 -- A  FUNCTION TGridView.VPointToCell (aPoint: VPoint): GridCell;
10 2073 --      VAR
10 2074 --          i:      INTEGER;
10 2075 --          aCell:  GridCell;
10 2076 0- A  BEGIN
10 2077 --          (* Decide what cell is indicated by this point. *)
10 2078 --          IF aPoint.h >= fMaxHorizontal THEN

```

```

10 2079 --      aPoint.h := fMaxHorizontal - 1;
10 2080 --
10 2081 --      IF aPoint.v >= fMaxVertical THEN
10 2082 --          aPoint.v := fMaxVertical - 1;
10 2083 --
10 2084 --      IF fNumOfColChunks = 1 THEN      (* only one column height *)
10 2085 1-      BEGIN
10 2086 --          IF fColWidths^[0].size = 0 THEN
10 2087 --              aCell.h := 1
10 2088 --          ELSE
10 2089 --              aCell.h := (aPoint.h DIV fColWidths^[0].size) + 1;
10 2090 -1      END
10 2091 --      ELSE
10 2092 1-      BEGIN
10 2093 --          i := 0;
10 2094 --          aCell.h := 0;
10 2095 --          aPoint.h := aPoint.h + 1;
10 2096 --          WHILE aPoint.h > 0 DO
10 2097 2-      .BEGIN
10 2098 --              aPoint.h := aPoint.h -
10 2099 --                  (fColWidths^[i].size * fColWidths^[i].count);
10 2100 --              aCell.h := aCell.h + fColWidths^[i].count;
10 2101 --              i := i + 1;
10 2102 -2      END;
10 2103 --          IF fColWidths^[i-1].size <> 0 THEN
10 2104 --              aCell.h := aCell.h - (ABS (aPoint.h) DIV fColWidths^[i-1].size);
10 2105 -1      END;
10 2106 --
10 2107 --      IF fNumOfRowChunks = 1 THEN      (* only one row height *)
10 2108 1-      BEGIN
10 2109 --          IF fRowHeights^[0].size = 0 THEN
10 2110 --              aCell.v := 1
10 2111 --          ELSE
10 2112 --              aCell.v := (aPoint.v DIV fRowHeights^[0].size) + 1;
10 2113 -1      END
10 2114 --      ELSE
10 2115 1-      BEGIN
10 2116 --          i := 0;
10 2117 --          aCell.v := 0;
10 2118 --          aPoint.v := aPoint.v + 1;
10 2119 --          WHILE aPoint.v > 0 DO
10 2120 2-      .BEGIN
10 2121 --              aPoint.v := aPoint.v -
10 2122 --                  (fRowHeights^[i].size * fRowHeights^[i].count);
10 2123 --              aCell.v := aCell.v + fRowHeights^[i].count;
10 2124 --              i := i + 1;
10 2125 -2      END;
10 2126 --          IF fRowHeights^[i-1].size <> 0 THEN
10 2127 --              aCell.v := aCell.v - (ABS (aPoint.v) DIV fRowHeights^[i-1].size);
10 2128 -1      END;
10 2129 --          VPointToCell := aCell;
10 2130 -0 A      END; (* TGridView.VPointToCell *)
10 2131 --
10 2132 --
10 2133 --      {*****}
10 2134 --      { * TGridView.Fields - for DEBUGGING * }
10 2135 --      {*****}
10 2136 --      {IFC qDebug}
10 2137 --      {SS GVFields}
10 2138 -- A      PROCEDURE TGridView.Fields (PROCEDURE DoToField (fieldName: Str255;
10 2139 --                                     fieldAddr: Ptr;
10 2140 --                                     fieldType: INTEGER)); OVERRIDE;
10 2141 0- A      BEGIN
10 2142 --          DoToField ('TGridView', NIL, bClass);
10 2143 --          DoToField ('fSelections', @fSelections, bHandle);
10 2144 --          DoToField ('fHLRegion', @fHLRegion, bHandle);
10 2145 --          DoToField ('fTmpSelRgn', @fTmpSelRgn, bHandle);
10 2146 --          DoToField ('fTmpRgn', @fTmpRgn, bHandle);
10 2147 --          DoToField ('fTmpHLRgn', @fTmpHLRgn, bHandle);
10 2148 --          DoToField ('fNumOfRows', @fNumOfRows, bInteger);
10 2149 --          DoToField ('fRowHeights', @fRowHeights, bHandle);
10 2150 --          DoToField ('fNumOfCols', @fNumOfCols, bInteger);
10 2151 --          DoToField ('fColWidths', @fColWidths, bHandle);
10 2152 --          DoToField ('fMaxHorizontal', @fMaxHorizontal, bLongInt);
10 2153 --          DoToField ('fMaxVertical', @fMaxVertical, bLongInt);
10 2154 --          DoToField ('fRowInset', @fRowInset, bInteger);
10 2155 --          DoToField ('fColInset', @fColInset, bInteger);
10 2156 --          DoToField ('fAdornRows', @fAdornRows, bBoolean);
10 2157 --          DoToField ('fAdornCols', @fAdornCols, bBoolean);
10 2158 --          DoToField ('fSingleSelection', @fSingleSelection, bBoolean);
10 2159 --          DoToField ('fAllowDimSelection', @fAllowDimSelection, bBoolean);
10 2160 --          DoToField ('fWholeRect', @fWholeRect, bBoolean);
10 2161 --          DoToField ('fLastWholeRect', @fLastWholeRect, bBoolean);
10 2162 --          DoToField ('fNumOfRowChunks', @fNumOfRowChunks, bInteger);
10 2163 --          DoToField ('fLastRow', @fLastRow, bInteger);
10 2164 --          DoToField ('fLastRowChunkNum', @fLastRowChunkNum, bInteger);
10 2165 --          DoToField ('fLastRowTotal', @fLastRowTotal, bLongInt);
10 2166 --          DoToField ('fLastRowIndex', @fLastRowIndex, bInteger);
10 2167 --          DoToField ('fNumOfColChunks', @fNumOfColChunks, bInteger);
10 2168 --          DoToField ('fLastCol', @fLastCol, bInteger);
10 2169 --          DoToField ('fLastColChunkNum', @fLastColChunkNum, bInteger);
10 2170 --          DoToField ('fLastColTotal', @fLastColTotal, bLongInt);
10 2171 --          DoToField ('fLastColIndex', @fLastColIndex, bInteger);
10 2172 --
10 2173 --          INHERITED Fields (DoToField);
10 2174 -0 A      END; (* TGridView.Fields *)
10 2175 --      {$ENDC qDebug}
10 2176 --
10 2177 --

```

```

10 2178 -- {*****}
10 2179 -- {*****}
10 2180 -- (* TTextView Object Sub-Class *)
10 2181 -- (* This sub-class implements a text-only version of the gridview. The *)
10 2182 -- (* user must override the GetText routine and we'll take care of the rest! *)
10 2183 -- (* TTextView calculates the height and width of rows and columns *)
10 2184 -- (* based on the font and the string length. Right now, the text in a cell *)
10 2185 -- (* can be only one line. (OVERRIDE the DrawCell method to do something a *)
10 2186 -- (* little slicker.) If you don't want the original height or width, *)
10 2187 -- (* individual rows and columns can be changed by calling SetColWidth and *)
10 2188 -- (* SetRowHeight. *)
10 2189 -- {*****}
10 2190 -- {*****}
10 2191 --
10 2192 --
10 2193 -- {*****}
10 2194 -- (* TTextView.ITextView *)
10 2195 -- (* Initialize the TTextView Object. *)
10 2196 -- {*****}
10 2197 -- ($IFC qProceduralViews)
10 2198 -- ($S GVOpen)
10 2199 -- A PROCEDURE TTextView.ITextView (
10 2200 --     itsDocument: TDocument; (* Its document *)
10 2201 --     itsSuperView: TView; (* Its parent view *)
10 2202 --     itsLocation: VPoint; (* Top, Left in parent's coords *)
10 2203 --     numofRows: INTEGER; (* Number of rows initially *)
10 2204 --     numofCols: INTEGER; (* Number of columns initially *)
10 2205 --     colWidth: INTEGER; (* Width of items in the columns *)
10 2206 --     adornRows: BOOLEAN; (* Adornment for Rows? *)
10 2207 --     adornCols: BOOLEAN; (* Adornment for Columns? *)
10 2208 --     rowInset: INTEGER; (* horizontal space between cells *)
10 2209 --     colInset: INTEGER; (* vertical space between cells *)
10 2210 --     singleSelection: BOOLEAN; (* single cell selection? *)
10 2211 --     itsTextStyle: TextStyle; (* size, color, etc. font info *)
10 2212 -- A BEGIN
10 2213 --     fTextStyle := itsTextStyle;
10 2214 --
10 2215 --     SetUpFont;
10 2216 --
10 2217 --     IGridView (itsDocument, itsSuperView, itsLocation, numofRows, numofCols,
10 2218 --         fLineHeight + rowInset, colWidth, adornRows, adornCols,
10 2219 --         rowInset, colInset, singleSelection);
10 2220 --
10 2221 --     IF (fNumofCols = 1) & (fSizeDeterminer[h] <> sizeFixed) &
10 2222 --         (GetColWidth (1) = 0) & (fSuperView <> NIL) THEN
10 2223 --         SetColWidth (1, fNumofCols, fSuperView.fSize.h);
10 2224 -- A END; (* TTextView.ITextView *)
10 2225 -- ($ENDC qProceduralViews)
10 2226 --
10 2227 --
10 2228 -- {*****}
10 2229 -- (* TTextView.IRes *)
10 2230 -- (* Initialize the GridView for use with view templates. *)
10 2231 -- {*****}
10 2232 -- ($IFC qTemplateViews)
10 2233 -- ($S GVOpen)
10 2234 -- A PROCEDURE TTextView.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
10 2235 -- VAR
10 2236 --     itsTextStyle: TextStyle;
10 2237 --
10 2238 -- A BEGIN
10 2239 --     INHERITED IRes (itsDocument, itsSuperView, itsParams);
10 2240 --
10 2241 --     WITH TextGridViewTemplatePtr (itsParams)^ DO
10 2242 --     BEGIN
10 2243 --
10 2244 --         SetTextStyle (itsTextStyle, GetFontNum (itsFontName), itsFontFace,
10 2245 --             itsFontSize, itsFontColor);
10 2246 --
10 2247 --         fTextStyle := itsTextStyle;
10 2248 --         SetUpFont;
10 2249 --     END;
10 2250 --
10 2251 --     IF fNumofRows > 0 THEN
10 2252 --         SetRowHeight (1, fNumofRows, fLineHeight + fRowInset);
10 2253 --
10 2254 --     IF (fNumofCols = 1) & (fSizeDeterminer[h] <> sizeFixed) & (GetColWidth (1) = 0) THEN
10 2255 --     BEGIN
10 2256 --         SetColWidth (1, fNumofCols, fSize.h);
10 2257 --     END;
10 2258 --
10 2259 --     OffsetPtrWStr (itsParams, SIZEOF (TextGridViewTemplate));
10 2260 -- A END; (* TTextView.IRes *)
10 2261 -- ($ENDC qTemplateViews)
10 2262 --
10 2263 --
10 2264 -- {*****}
10 2265 -- (* TTextView.WRes *)
10 2266 -- (* Write the resource file. *)
10 2267 -- {*****}
10 2268 -- ($IFC qWriteTemplates)
10 2269 -- ($S GVNonRes)
10 2270 -- A PROCEDURE TTextView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
10 2271 --
10 2272 -- VAR
10 2273 --     theSize: INTEGER;
10 2274 --     theFont: Str255;
10 2275 --     tgPtr: TextGridViewTemplatePtr;
10 2276 --

```

```

10 2277 0- A BEGIN
10 2278 --      INHERITED WRes(theResource, itsParams);
10 2279 --
10 2280 --      theSize := fTextStyle.tsSize;
10 2281 --      GetPortFontInfo(fTextStyle.tsFont, theFont, theSize);
10 2282 --
10 2283 --      tgPtr := TextGridViewTemplatePtr(ExpandPtrWStr(theResource, itsParams,
10 2284 --      SIZEOF(TextGridViewTemplate), LENGTH(theFont)));
10 2285 --
10 2286 --      WITH tgPtr^, fTextStyle DO
10 2287 1-      BEGIN
10 2288 --          itsFontFace := tsFace;
10 2289 --          itsFontSize := theSize;
10 2290 --          itsFontColor := tsColor;
10 2291 --          (* itsFontName := theFont; *)
10 2292 -1      END;
10 2293 --      CopyStr255(theFont, PRStr(tgPtr^.itsFontName));
10 2294 -0 A END; (* TTextGridView.WRes *)
10 2295 --
10 2296 --
10 2297 --      {*****}
10 2298 --      (* TTextGridView.WriteRes *)
10 2299 --      (* Set up for writing the resource file *)
10 2300 --      {*****}
10 2301 --      {$S GVNonRes}
10 2302 -- A PROCEDURE TTextGridView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
10 2303 0- A BEGIN
10 2304 --      gWResSignature := 'txtg'; gWResType := 'TTextGridView';
10 2305 --      WRes(theResource, itsParams);
10 2306 -0 A END; (* TTextGridView.WriteRes *)
10 2307 --      {$ENDC qWriteTemplates}
10 2308 --
10 2309 --
10 2310 --      {*****}
10 2311 --      (* TTextGridView.DrawCell *)
10 2312 --      (* Get the text that needs to be drawn and draw it. *)
10 2313 --      {*****}
10 2314 --      {$S GVRes}
10 2315 -- A PROCEDURE TTextGridView.DrawCell(aCell: GridCell; aQDRect: Rect); OVERRIDE;
10 2316 --      VAR
10 2317 --          theText: Str255;
10 2318 0- A BEGIN
10 2319 --          GetText (aCell, theText);
10 2320 --
10 2321 --          IF (GetColWidth (aCell.h) > 0) THEN
10 2322 1-      BEGIN
10 2323 --              MoveTo (aQDRect.left, aQDRect.top + fLineAscent);
10 2324 --              DrawString (theText);
10 2325 -1      END;
10 2326 -0 A END; (* TTextGridView.DrawCell *)
10 2327 --
10 2328 --
10 2329 --      {*****}
10 2330 --      (* TTextGridView.Focus *)
10 2331 --      (* Set the pen and font when focus'ing. *)
10 2332 --      {*****}
10 2333 --      {$S GVRes}
10 2334 -- A FUNCTION TTextGridView.Focus : BOOLEAN; OVERRIDE;
10 2335 0- A BEGIN
10 2336 --      IF INHERITED Focus THEN
10 2337 1-      BEGIN
10 2338 --          SetPen;
10 2339 --          Focus := TRUE;
10 2340 -1      END
10 2341 --      ELSE
10 2342 --          Focus := FALSE;
10 2343 -0 A END; (* TTextGridView.Focus *)
10 2344 --
10 2345 --
10 2346 --      {*****}
10 2347 --      (* TTextGridView.SetUpFont *)
10 2348 --      (* Sets the font for this view. *)
10 2349 --      {*****}
10 2350 --      {$S GVOpen}
10 2351 -- A PROCEDURE TTextGridView.SetUpFont;
10 2352 --      VAR
10 2353 --          theFontInfo: FontInfo;
10 2354 --
10 2355 0- A BEGIN
10 2356 --          SetPen;
10 2357 --          GetFontInfo (theFontInfo);
10 2358 --          WITH theFontInfo DO
10 2359 1-      BEGIN
10 2360 --              fLineHeight := ascent + descent + leading;
10 2361 --              fLineAscent := ascent + (leading DIV 2);
10 2362 -1      END;
10 2363 -0 A END; (* TTextGridView.SetUpFont *)
10 2364 --
10 2365 --
10 2366 --      {*****}
10 2367 --      (* TTextGridView.SetPen *)
10 2368 --      (* Sets the font characteristics of the current port. Called at *)
10 2369 --      (* the beginning of the Draw method. *)
10 2370 --      {*****}
10 2371 --      {$S GVRes}
10 2372 -- A PROCEDURE TTextGridView.SetPen;
10 2373 --      VAR
10 2374 --          itsTextStyle: TextStyle;
10 2375 --

```



```

10 2376 0- A BEGIN
10 2377 --      itsTextStyle := fTextStyle;
10 2378 --      SetPortTextStyle (itsTextStyle);
10 2379 --      PenNormal;
10 2380 0- A END; (* TTextGridView.SetPen *)
10 2381 --
10 2382 --
10 2383 --      {*****}
10 2384 --      (* TTextGridView.GetText *)
10 2385 --      (* This method gets the text to be drawn in a given cell. IT MUST BE *)
10 2386 --      (* OVERRIDDEN !! *)
10 2387 --      {*****}
10 2388 --      {$S GVRes}
10 2389 -- A PROCEDURE TTextGridView.GetText (aCell: GridCell; VAR aString: Str255);
10 2390 0- A BEGIN
10 2391 --      (* MUST be overridden! *)
10 2392 --
10 2393 --      {$IFC qDebug}
10 2394 --      WRITELN ('TTextGridView: GetText MUST be OVERRIDDEN!');
10 2395 --      {$ENDC qDebug}
10 2396 0- A END; (* TTextGridView.GetText *)
10 2397 --
10 2398 --
10 2399 --      {*****}
10 2400 --      (* TTextGridView.Fields *)
10 2401 --      (* For debugging support of the inspector. *)
10 2402 --      {*****}
10 2403 --      {$IFC qDebug}
10 2404 --      {$S GVFields}
10 2405 -- A PROCEDURE TTextGridView.Fields ( PROCEDURE DoToField (fieldName: Str255;
10 2406 --      fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 2407 0- A BEGIN
10 2408 --      DoToField ('TTextGridView', NIL, bClass);
10 2409 --      DoToField ('fLineHeight', @fLineHeight, bInteger);
10 2410 --      DoToField ('fLineAscent', @fLineAscent, bInteger);
10 2411 --      {$H-}
10 2412 --      TextStyleFields ('fTextStyle', fTextStyle, DoToField);
10 2413 --      {$H+}
10 2414 --
10 2415 --      INHERITED Fields (DoToField);
10 2416 0- A END; (* TTextGridView.Fields *)
10 2417 --      {$ENDC qDebug}
10 2418 --
10 2419 --
10 2420 --      {*****}
10 2421 --      {*****}
10 2422 --      (* TTextListView Object Sub-Class *)
10 2423 --      (* This sub-class implements a single column version of TextGridView. The *)
10 2424 --      (* user must override the GetItemText routine and the rest is done! *)
10 2425 --      (* TTextListView calculates the height and width of items *)
10 2426 --      (* based on the font and the string length. Right now, the text in a cell *)
10 2427 --      (* can be only one line. (OVERRIDE the DrawCell method to do something a *)
10 2428 --      (* little slicker.) If you don't want the original height, individual *)
10 2429 --      (* items can be changed by calling SetItemHeight. The width of view can *)
10 2430 --      (* be modified by calling SetItemWidth. *)
10 2431 --      {*****}
10 2432 --      {*****}
10 2433 --
10 2434 --
10 2435 --      {*****}
10 2436 --      (* TTextListView.ITextListView *)
10 2437 --      (* Initialize the TextListView object. *)
10 2438 --      {*****}
10 2439 --      {$IFC qProceduralViews}
10 2440 --      {$S GVOpen}
10 2441 -- A PROCEDURE TTextListView.ITextListView (
10 2442 --      itsDocument: TDocument; (* Its document *)
10 2443 --      itsSuperView: TView; (* Its parent view *)
10 2444 --      itsLocation: VPoint; (* Top, Left in parent's coords *)
10 2445 --      numOfItems: INTEGER; (* Number of items initially *)
10 2446 --      adornRows: BOOLEAN; (* Draw the row adornments? *)
10 2447 --      adornCols: BOOLEAN; (* Draw the col adornment? *)
10 2448 --      rowInset: INTEGER; (* Amount to inset the rows *)
10 2449 --      colInset: INTEGER; (* Amount to inset the column *)
10 2450 --      singleSelection: BOOLEAN; (* single cell selection? *)
10 2451 --      itsTextStyle: TextStyle; (* size, color, etc. font info *)
10 2452 --
10 2453 --      CONST
10 2454 --      kOneColumn = 1;
10 2455 --
10 2456 0- A BEGIN
10 2457 --      ITextGridView (itsDocument, itsSuperView, itsLocation, numOfItems, kOneColumn,
10 2458 --      0, adornRows, adornCols, rowInset, colInset, singleSelection,
10 2459 --      itsTextStyle);
10 2460 --
10 2461 --      fSizeDeterminer[h] := sizeRelSuperView;
10 2462 0- A END; (* TTextListView.ITextListView *)
10 2463 --      {$ENDC qProceduralViews}
10 2464 --
10 2465 --
10 2466 --      {*****}
10 2467 --      (* TTextListView.WriteRes *)
10 2468 --      (* Set up for writing the resource file *)
10 2469 --      {*****}
10 2470 --      {$IFC qWriteTemplates}
10 2471 --      {$S GVNonRes}
10 2472 -- A PROCEDURE TTextListView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
10 2473 0- A BEGIN
10 2474 --      gWRSignature := 'lstg'; gWRType := 'TTextListView';

```

```

10 2475 --      WRes(theResource, itsParams);
10 2476 -- A   END: (* TTextListView.WriteRes *)
10 2477 --      ($ENDC qWriteTemplates)
10 2478 --
10 2479 --
10 2480 --      {*****}
10 2481 --      (* TTextListView.AllItemsDo *)
10 2482 --      (* For all the items execute "DoToItem". *)
10 2483 --      {*****}
10 2484 --      ($$ GVRes)
10 2485 -- A   PROCEDURE TTextListView.AllItemsDo (PROCEDURE DoToItem (anItem: INTEGER));
10 2486 -- A   BEGIN
10 2487 --      EachItemDo (1, fNumOfRows, DoToItem);
10 2488 -- A   END: (* TTextListView.AllItemsDo *)
10 2489 --
10 2490 --
10 2491 --      {*****}
10 2492 --      (* TTextListView.CanSelectCell *)
10 2493 --      (* Figure out if the cell can be selected. *)
10 2494 --      {*****}
10 2495 --      ($$ GVRes)
10 2496 -- A   FUNCTION TTextListView.CanSelectCell (aCell: GridCell) : BOOLEAN; OVERRIDE;
10 2497 -- A   BEGIN
10 2498 --      CanSelectCell := CanSelectItem (aCell.v);
10 2499 -- A   END: (* TTextListView.CanSelectCell *)
10 2500 --
10 2501 --
10 2502 --      {*****}
10 2503 --      (* TTextListView.CanSelectItem *)
10 2504 --      (* Figure out if the item can be selected. *)
10 2505 --      {*****}
10 2506 --      ($$ GVRes)
10 2507 -- A   FUNCTION TTextListView.CanSelectItem (anItem: INTEGER) : BOOLEAN;
10 2508 -- A   BEGIN
10 2509 --      CanSelectItem := TRUE;
10 2510 -- A   END: (* TTextListView.CanSelectItem *)
10 2511 --
10 2512 --
10 2513 --      {*****}
10 2514 --      (* TTextListView.DelItemAt *)
10 2515 --      (* Delete numOfItems starting at anItem. This method causes a *)
10 2516 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2517 --      (* (and adornments). *)
10 2518 --      {*****}
10 2519 --      ($$ GVNonRes)
10 2520 -- A   PROCEDURE TTextListView.DelItemAt (anItem: INTEGER; numOfItems: INTEGER);
10 2521 -- A   BEGIN
10 2522 --      DelRowAt (anItem, numOfItems);
10 2523 -- A   END: (* TTextListView.DelItemAt *)
10 2524 --
10 2525 --
10 2526 --      {*****}
10 2527 --      (* TTextListView.DelItemFirst *)
10 2528 --      (* Delete numOfItems at the beginning. This method causes a *)
10 2529 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2530 --      (* (and adornments). *)
10 2531 --      {*****}
10 2532 --      ($$ GVNonRes)
10 2533 -- A   PROCEDURE TTextListView.DelItemFirst (numOfItems: INTEGER);
10 2534 -- A   BEGIN
10 2535 --      DelItemAt (1, numOfItems);
10 2536 -- A   END: (* TTextListView.DelItemFirst *)
10 2537 --
10 2538 --
10 2539 --      {*****}
10 2540 --      (* TTextListView.DelItemLast *)
10 2541 --      (* Delete numOfItems at the end. This method causes a *)
10 2542 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2543 --      (* (and adornments). *)
10 2544 --      {*****}
10 2545 --      ($$ GVNonRes)
10 2546 -- A   PROCEDURE TTextListView.DelItemLast (numOfItems: INTEGER);
10 2547 -- A   BEGIN
10 2548 --      DelItemAt (fNumOfRows, numOfItems);
10 2549 -- A   END: (* TTextListView.DelItemLast *)
10 2550 --
10 2551 --
10 2552 --      {*****}
10 2553 --      (* TTextListView.EachItemDo *)
10 2554 --      (* For each item from start to stop execute "DoToItem". *)
10 2555 --      {*****}
10 2556 --      ($$ GVRes)
10 2557 -- A   PROCEDURE TTextListView.EachItemDo (start, stop: INTEGER;
10 2558 --      PROCEDURE DoToItem (anItem: INTEGER));
10 2559 -- A   VAR
10 2560 --      i: INTEGER;
10 2561 --
10 2562 -- A   BEGIN
10 2563 --      FOR i := start TO stop DO
10 2564 --          DoToItem (i);
10 2565 -- A   END: (* TTextListView.EachItemDo *)
10 2566 --
10 2567 --
10 2568 --      {*****}
10 2569 --      (* TTextListView.EachSelectedItemDo *)
10 2570 --      (* For each item from start to stop execute "DoToItem". *)
10 2571 --      {*****}
10 2572 --      ($$ GVRes)
10 2573 -- A   PROCEDURE TTextListView.EachSelectedItemDo (PROCEDURE DoToItem (anItem: INTEGER));

```

```

10 2574 -- B      PROCEDURE DoToCell (aCell: GridCell);
10 2575 0- B      BEGIN
10 2576 --      DoToItem (aCell.v);
10 2577 0- B      END;
10 2578 --
10 2579 0- A      BEGIN
10 2580 --          EachInRgn (fSelections, DoToCell);
10 2581 0- A      END; (* TTextListView.EachSelectedItemDo *)
10 2582 --
10 2583 --
10 2584 --      {*****}
10 2585 --      (* TTextListView.GetItemHeight *)
10 2586 --      (* Get the height of the specified item. *)
10 2587 --      {*****}
10 2588 --      {$S GVRes}
10 2589 -- A      FUNCTION TTextListView.GetItemHeight (anItem: INTEGER) : INTEGER;
10 2590 0- A      BEGIN
10 2591 --          GetItemHeight := GetRowHeight (anItem);
10 2592 0- A      END; (* TTextListView.GetItemHeight *)
10 2593 --
10 2594 --
10 2595 --      {*****}
10 2596 --      (* TTextListView.GetItemWidth *)
10 2597 --      (* Get the width of the view. (All the items have the same width!) *)
10 2598 --      {*****}
10 2599 --      {$S GVRes}
10 2600 -- A      FUNCTION TTextListView.GetItemWidth: INTEGER;
10 2601 0- A      BEGIN
10 2602 --          GetItemWidth := GetColWidth (1);
10 2603 0- A      END; (* TTextListView.GetItemWidth *)
10 2604 --
10 2605 --
10 2606 --      {*****}
10 2607 --      (* TTextListView.GetItemText *)
10 2608 --      (* Get the text for the specified item. THIS METHOD MUST BE OVERRIDDEN!! *)
10 2609 --      {*****}
10 2610 --      {$S GVRes}
10 2611 -- A      PROCEDURE TTextListView.GetItemText (anItem: INTEGER; VAR aString: Str255);
10 2612 0- A      BEGIN
10 2613 --          (* MUST be overridden !!! *)
10 2614 --
10 2615 --      ($IFC qDebug)
10 2616 --          WRITELN ('TTextListView: GetItemText MUST be OVERRIDDEN!');
10 2617 --      ($ENDC qDebug)
10 2618 --
10 2619 0- A      END; (* TTextListView.GetItemText *)
10 2620 --
10 2621 --
10 2622 --      {*****}
10 2623 --      (* TTextListView.GetText *)
10 2624 --      (* Figure out which item it is and get the text. *)
10 2625 --      {*****}
10 2626 --      {$S GVRes}
10 2627 -- A      PROCEDURE TTextListView.GetText (aCell: GridCell; VAR aString: Str255); OVERRIDE;
10 2628 0- A      BEGIN
10 2629 --          GetItemText (aCell.v, aString);
10 2630 0- A      END; (* TTextListView.GetText *)
10 2631 --
10 2632 --
10 2633 --      {*****}
10 2634 --      (* TTextListView.InsItemBefore *)
10 2635 --      (* Insert numOfItems starting at anItem. This method causes a *)
10 2636 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2637 --      (* (and adornments). *)
10 2638 --      {*****}
10 2639 --      {$S GVRes}
10 2640 -- A      PROCEDURE TTextListView.InsItemBefore (anItem: INTEGER; numOfItems: INTEGER);
10 2641 0- A      BEGIN
10 2642 --          InsRowBefore (anItem, numOfItems, fLineHeight + fRowInset);
10 2643 0- A      END; (* TTextListView.InsItemBefore *)
10 2644 --
10 2645 --
10 2646 --      {*****}
10 2647 --      (* TTextListView.InsItemFirst *)
10 2648 --      (* Insert numOfItems at the beginning. This method causes a *)
10 2649 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2650 --      (* (and adornments). *)
10 2651 --      {*****}
10 2652 --      {$S GVRes}
10 2653 -- A      PROCEDURE TTextListView.InsItemFirst (numOfItems: INTEGER);
10 2654 0- A      BEGIN
10 2655 --          InsItemBefore (1, numOfItems);
10 2656 0- A      END; (* TTextListView.InsItemFirst *)
10 2657 --
10 2658 --
10 2659 --      {*****}
10 2660 --      (* TTextListView.InsItemLast *)
10 2661 --      (* Insert numOfItems at the end. This method causes a *)
10 2662 --      (* re-calculation of the extent and redraws only the necessary cells *)
10 2663 --      (* (and adornments). *)
10 2664 --      {*****}
10 2665 --      {$S GVRes}
10 2666 -- A      PROCEDURE TTextListView.InsItemLast (numOfItems: INTEGER);
10 2667 0- A      BEGIN
10 2668 --          InsItemBefore (fNumOfRows + 1, numOfItems);
10 2669 0- A      END; (* TTextListView.InsItemLast *)
10 2670 --
10 2671 --
10 2672 --      {*****}

```

```

10 2673 -- (* TTextView.IsItemSelected *)
10 2674 -- (* Check if the specified item is currently selected. *)
10 2675 -- {*****}
10 2676 -- {$S GVRes}
10 2677 -- A FUNCTION TTextView.IsItemSelected (anItem: INTEGER) : BOOLEAN;
10 2678 -- VAR
10 2679 --     aCell: GridCell;
10 2680 --
10 2681 0- A BEGIN
10 2682 --     aCell.h := 1;
10 2683 --     aCell.v := anItem;
10 2684 --     IsItemSelected := IsCellSelected (aCell);
10 2685 -0 A END; (* TTextView.IsItemSelected *)
10 2686 --
10 2687 --
10 2688 -- {*****}
10 2689 -- (* TTextView.Resize *)
10 2690 -- (* Resize the view. *)
10 2691 -- {*****}
10 2692 -- {$S GVNonRes}
10 2693 -- A PROCEDURE TTextView.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
10 2694 0- A BEGIN
10 2695 --     INHERITED Resize (width, height, invalidate);
10 2696 --     IF fNumOfCols = 1 THEN
10 2697 1- BEGIN
10 2698 --         fMaxHorizontal := fMaxHorizontal - fColWidths^[0].size + width;
10 2699 --         fColWidths^[0].size := width; { We know there's only 1 column }
10 2700 -1 END;
10 2701 -0 A END; (* TTextView.Resize *)
10 2702 --
10 2703 --
10 2704 -- {*****}
10 2705 -- (* TTextView.SelectCell *)
10 2706 -- (* Get the correct item number and call SelectItem. *)
10 2707 -- {*****}
10 2708 -- {$S GVRes}
10 2709 -- A PROCEDURE TTextView.SelectCell (theCell: GridCell;
10 2710 --     extend, highlight, select: BOOLEAN); OVERRIDE;
10 2711 0- A BEGIN
10 2712 --     SelectItem (theCell.v, extend, highlight, select);
10 2713 -0 A END; (* TTextView.SelectCell *)
10 2714 --
10 2715 --
10 2716 -- {*****}
10 2717 -- (* TTextView.SelectItem *)
10 2718 -- (* Selects a single item in the view. *)
10 2719 -- {*****}
10 2720 -- {$S GVRes}
10 2721 -- A PROCEDURE TTextView.SelectItem (anItem: INTEGER; extend, highlight, select: BOOLEAN);
10 2722 -- VAR
10 2723 --     aCell: GridCell;
10 2724 0- A BEGIN
10 2725 --     aCell.h := 1;
10 2726 --     aCell.v := anItem;
10 2727 --
10 2728 --     INHERITED SelectCell (aCell, extend, highlight, select);
10 2729 -0 A END; (* TTextView.SelectItem *)
10 2730 --
10 2731 --
10 2732 -- {*****}
10 2733 -- (* TTextView.SetItemHeight *)
10 2734 -- (* Set the height of the specified item. *)
10 2735 -- {*****}
10 2736 -- {$S GVNonRes}
10 2737 -- A PROCEDURE TTextView.SetItemHeight (anItem: INTEGER; numOfItems: INTEGER;
10 2738 --     aHeight: INTEGER);
10 2739 0- A BEGIN
10 2740 --     SetRowHeight (anItem, numOfItems, aHeight);
10 2741 -0 A END; (* TTextView.SetItemHeight *)
10 2742 --
10 2743 --
10 2744 -- {*****}
10 2745 -- (* TTextView.SetItemWidth *)
10 2746 -- (* Set the width of the view. (All the items have the same width!) *)
10 2747 -- {*****}
10 2748 -- {$S GVNonRes}
10 2749 -- A PROCEDURE TTextView.SetItemWidth (aWidth: INTEGER);
10 2750 0- A BEGIN
10 2751 --     SetColWidth (1, 1, aWidth);
10 2752 -0 A END; (* TTextView.SetItemWidth *)
10 2753 --
10 2754 --
10 2755 -- {*****}
10 2756 -- (* TTextView.Fields *)
10 2757 -- (* Debugging support for the inspector. *)
10 2758 -- {*****}
10 2759 -- {$IFC qDebug}
10 2760 -- {$S GVFields}
10 2761 -- A PROCEDURE TTextView.Fields (PROCEDURE DoToField (fieldName: Str255;
10 2762 --     fieldAddr: Ptr; fieldType: INTEGER); OVERRIDE;
10 2763 0- A BEGIN
10 2764 --     DoToField ('TTextView', NIL, bClass);
10 2765 --     INHERITED Fields (DoToField);
10 2766 -0 A END; (* TTextView.Fields *)
10 2767 -- {$ENDC qDebug}
10 2768 --
10 2769 --
10 2770 -- {*****}
10 2771 -- (* TGridSelectCommand *)

```

```

10 2772 -- {*****}
10 2773 -- {*****}
10 2774 -- (* TGridSelectCommand.IGridSelectCommand *)
10 2775 -- (* Skeleton object used to build needed command objects. *)
10 2776 -- {*****}
10 2777 -- {$S GVOpen}
10 2778 -- A PROCEDURE TGridSelectCommand.IGridSelectCommand (itsCmdNumber: CmdNumber;
10 2779 -- itsView: TGridView; theShiftKey, theCmdKey: BOOLEAN);
10 2780 -- CONST
10 2781 -- kUseImmediateSuperView = TRUE;
10 2782 -- VAR
10 2783 -- tmpRgn: RgnHandle;
10 2784 -- A BEGIN
10 2785 -- ICommand (itsCmdNumber, NIL, itsView, itsView.GetScroller (kUseImmediateSuperView));
10 2786 -- fCanUndo := FALSE;
10 2787 -- fCausesChange := FALSE;
10 2788 --
10 2789 -- fShiftKey := theShiftKey;
10 2790 -- fCmdKey := theCmdKey;
10 2791 -- fGridView := itsView;
10 2792 -- fConstrainsMouse := FALSE;
10 2793 --
10 2794 -- tmpRgn := MakeNewRgn;
10 2795 -- fOldRgn := tmpRgn; (* temporary region for old selection *)
10 2796 -- tmpRgn := MakeNewRgn;
10 2797 -- fNewRgn := tmpRgn; (* temporary region for new selection *)
10 2798 -- A END; (* TGridSelectCommand.IGridSelectCommand *)
10 2799 --
10 2800 --
10 2801 -- {*****}
10 2802 -- (* TGridSelectCommand.Free *)
10 2803 -- (* Free the temporary regions. *)
10 2804 -- {*****}
10 2805 -- {$S GVClose}
10 2806 -- A PROCEDURE TGridSelectCommand.Free;
10 2807 -- A BEGIN
10 2808 -- DisposeIfHandle (fOldRgn);
10 2809 -- DisposeIfHandle (fNewRgn);
10 2810 --
10 2811 -- INHERITED Free;
10 2812 -- A END; (* TGridSelectCommand.Free *)
10 2813 --
10 2814 --
10 2815 -- {*****}
10 2816 -- (* TGridSelectCommand.TrackFeedback *)
10 2817 -- (* Track the mouse movements when the mouse button is down. *)
10 2818 -- {*****}
10 2819 -- {$S GVRes}
10 2820 -- A PROCEDURE TGridSelectCommand.TrackFeedback (anchorPoint, nextPoint: VPoint;
10 2821 -- turnItOn, mouseDidMove: BOOLEAN); OVERRIDE;
10 2822 -- A BEGIN
10 2823 -- A END; (* TGridSelectCommand.TrackFeedback *)
10 2824 --
10 2825 --
10 2826 -- {*****}
10 2827 -- (* TGridSelectCommand.Fields *)
10 2828 -- (* For Debugging support of the inspector. *)
10 2829 -- {*****}
10 2830 -- {$IFC qDebug}
10 2831 -- {$S GVFields}
10 2832 -- A PROCEDURE TGridSelectCommand.Fields ( PROCEDURE DoToField (fieldName: Str255;
10 2833 -- fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 2834 -- A BEGIN
10 2835 -- DoToField ('TGridSelectCommand', NIL, bClass);
10 2836 -- DoToField ('fGridView', @fGridView, bObject);
10 2837 -- DoToField ('fShiftKey', @fShiftKey, bBoolean);
10 2838 -- DoToField ('fCmdKey', @fCmdKey, bBoolean);
10 2839 --
10 2840 -- INHERITED Fields (DoToField);
10 2841 -- A END; (* TGridSelectCommand.Fields *)
10 2842 -- {$ENDC qDebug}
10 2843 --
10 2844 --
10 2845 -- {*****}
10 2846 -- (* TCellSelectCommand *)
10 2847 -- {*****}
10 2848 -- {*****}
10 2849 -- (* TCellSelectCommand.ICellSelectCommand *)
10 2850 -- (* Initialize the cell select command object. *)
10 2851 -- {*****}
10 2852 -- {$S GVOpen}
10 2853 -- A PROCEDURE TCellSelectCommand.ICellSelectCommand (itsView: TGridView;
10 2854 -- aCell: GridCell; theShiftKey, theCmdKey: BOOLEAN);
10 2855 -- A BEGIN
10 2856 -- IGridSelectCommand (cCellSelect, itsView, theShiftKey, theCmdKey);
10 2857 -- fNewMouseDown := TRUE;
10 2858 -- A END; (* TCellSelectCommand.ICellSelectCommand *)
10 2859 --
10 2860 --
10 2861 -- {*****}
10 2862 -- (* TCellSelectCommand.HandleShiftKeyDown *)
10 2863 -- (* This routine provides mouse down support for TCellSelectCommand *)
10 2864 -- (* TrackFeedback and TrackMouse. *)
10 2865 -- (* It implements the functionality of the List Manager's cell selection *)
10 2866 -- (* algorithm. (See Inside Macintosh IV for more details.) *)
10 2867 -- {*****}
10 2868 -- {$S GVRes}
10 2869 -- A PROCEDURE TCellSelectCommand.HandleShiftKeyDown (aView: TGridView;
10 2870 -- aCell, anchorCell: GridCell; mouseDidMove: BOOLEAN);

```

```

10 2871 --      VAR
10 2872 --          boundRect: Rect;    (* Bounding Rect of the region *)
10 2873 --          newRect:   Rect;    (* new rectangle to highlight *)
10 2874 --          newQDRect:  Rect;
10 2875 --          oldQDRect:  Rect;
10 2876 --          newVRect:   VRect;
10 2877 --          oldVRect:   VRect;
10 2878 --          newBRRRect: VRect;
10 2879 --          oldBRRRect: VRect;
10 2880 --
10 2881 0- A  BEGIN
10 2882 --      WITH aView DO
10 2883 1-      BEGIN
10 2884 --          boundRect := fHLRegion^.rgnBBox;
10 2885 --          boundRect.bottom := boundRect.bottom - 1;
10 2886 --          boundRect.right  := boundRect.right - 1;
10 2887 --
10 2888 --          IF fNewMouseDown THEN
10 2889 2-      BEGIN (* shift click *)
10 2890 --              newRect.top    := MIN (aCell.v, boundRect.top);
10 2891 --              newRect.left   := MIN (aCell.h, boundRect.left);
10 2892 --              newRect.bottom := MAX (aCell.v, boundRect.top) + 1;
10 2893 --              newRect.right  := MAX (aCell.h, boundRect.left) + 1;
10 2894 -2      END
10 2895 --          ELSE
10 2896 2-      BEGIN (* dragging *)
10 2897 --              newRect.top    := MIN (aCell.v, anchorCell.v);
10 2898 --              newRect.left   := MIN (aCell.h, anchorCell.h);
10 2899 --              newRect.bottom := MAX (aCell.v, anchorCell.v) + 1;
10 2900 --              newRect.right  := MAX (aCell.h, anchorCell.h) + 1;
10 2901 -2      END;
10 2902 --
10 2903 --          (* Get the region to turn on *)
10 2904 --          RectRgn (fTmpRgn, newRect);
10 2905 --          CreateHighlightRgn (fTmpRgn, fNewRgn, kWholeRect);
10 2906 --
10 2907 --          (* Get the region to turn off *)
10 2908 --          CreateHighlightRgn (fHLRegion, fOldRgn, fWholeRect);
10 2909 --
10 2910 --          (* turn off the Old bits *)
10 2911 --          CopyRgn (fOldRgn, fTmpRgn);
10 2912 --          DiffRgn (fOldRgn, fNewRgn, fOldRgn);
10 2913 --          DoHilite (fOldRgn, hLOn, hLOff);
10 2914 --          DiffRgn (fNewRgn, fTmpRgn, fNewRgn);
10 2915 --
10 2916 --          (* turn on the new bits *)
10 2917 --          DoHilite (fNewRgn, hLOff, hLOn);
10 2918 --
10 2919 --          (* set fHLRegion to be correct *)
10 2920 --          RectRgn (fHLRegion, newRect);
10 2921 -1      END;
10 2922 -0 A  END; (* TCellSelectCommand.HandleShiftKeyDown *)
10 2923 --
10 2924 --
10 2925 --      {*****}
10 2926 --      (* TCellSelectCommand.TrackFeedback *)
10 2927 --      (* Track the mouse movements when the mouse button is down. *)
10 2928 --      {*****}
10 2929 --      {$S GVRes}
10 2930 -- A  PROCEDURE TCellSelectCommand.TrackFeedback (anchorPoint, nextPoint: VPoint;
10 2931 --      turnItOn, mouseDidMove: BOOLEAN); OVERRIDE;
10 2932 --
10 2933 --      VAR
10 2934 --          aCell:      GridCell;
10 2935 --          anchorCell: GridCell;
10 2936 --          aRect:      VRect;
10 2937 --          hlRect:      Rect;
10 2938 0- A  BEGIN
10 2939 --      WITH fGridView DO
10 2940 1-      BEGIN
10 2941 --          aCell := VPointToCell (nextPoint);
10 2942 --
10 2943 --          IF NOT (SameCell (aCell, fOldCell)) &
10 2944 --              (aCell.v > 0) & (aCell.h > 0) &
10 2945 --              (aCell.v <= fNumOfRows) & (aCell.h <= fNumOfCols) &
10 2946 --              mouseDidMove & CanSelectCell (aCell) THEN
10 2947 2-      BEGIN
10 2948 --          (* get the new selection *)
10 2949 --          CellToVRect (aCell, aRect);
10 2950 --          ViewToQDRect (aRect, hlRect);
10 2951 --          RectRgn (fNewRgn, hlRect);
10 2952 --
10 2953 --          (* set the highlighted region *)
10 2954 --          hlRect.topleft := aCell;
10 2955 --          hlRect.bottom  := aCell.v + 1;
10 2956 --          hlRect.right   := aCell.h + 1;
10 2957 --
10 2958 --          IF fCmdKey & NOT (fSingleSelection) THEN
10 2959 3-      BEGIN
10 2960 --          IF fNewMouseDown THEN
10 2961 4-      BEGIN
10 2962 --              (* turn cells on or off? *)
10 2963 --              IF PtInRgn (aCell, fHLRegion) THEN
10 2964 --                  fKeepSelecting := FALSE
10 2965 --              ELSE
10 2966 --                  fKeepSelecting := TRUE;
10 2967 --              fNewMouseDown := FALSE;
10 2968 -4      END;
10 2969 --          END;

```

```

10 2970 --      (* set up the selection region containing the cell *)
10 2971 --      RectRgn (fTmpRgn, h1Rect);
10 2972 --
10 2973 --      IF fKeepSelecting THEN
10 2974 4-      BEGIN
10 2975 --          IF NOT PtInRgn (aCell, fHLRegion) THEN
10 2976 5-      BEGIN
10 2977 --              UnionRgn (fHLRegion, fTmpRgn, fHLRegion);
10 2978 --              DoHilite (fNewRgn, h1OFF, h1ON);
10 2979 -5      END;
10 2980 -4      END
10 2981 --      ELSE IF PtInRgn (aCell, fHLRegion) THEN
10 2982 4-      BEGIN
10 2983 --          DiffRgn (fHLRegion, fTmpRgn, fHLRegion);
10 2984 --          DoHilite (fNewRgn, h1ON, h1OFF);
10 2985 -4      END;
10 2986 --
10 2987 --      fWholeRect := FALSE; (* the selected region may not be a rectangle *)
10 2988 -3      END
10 2989 --
10 2990 --      ELSE IF fShiftKey & NOT (fSingleSelection) THEN
10 2991 3-      BEGIN
10 2992 --          anchorCell := VPointToCell (anchorPoint);
10 2993 --          HandleShiftKeyDown (fGridView, aCell, anchorCell, mouseDidMove);
10 2994 --          fWholeRect := TRUE; (* the selected region is a complete rectangle *)
10 2995 -3      END
10 2996 --
10 2997 --      ELSE (* No recognized keys down with the mouse *)
10 2998 3-      BEGIN
10 2999 --          (* get the old selection *)
10 3000 --          CreateHighlightRgn (fHLRegion, fOldRgn, fWholeRect);
10 3001 --
10 3002 --          (* Turn off the old ones *)
10 3003 --          DiffRgn (fOldRgn, fNewRgn, fOldRgn);
10 3004 --          DoHilite (fOldRgn, h1ON, h1OFF);
10 3005 --
10 3006 --          (* highlight the new selection *)
10 3007 --          IF NOT PtInRgn (aCell, fHLRegion) THEN
10 3008 --              DoHilite (fNewRgn, h1OFF, h1ON);
10 3009 --
10 3010 --          RectRgn (fHLRegion, h1Rect);
10 3011 --
10 3012 --          fWholeRect := TRUE; (* the selected region is a rectangle *)
10 3013 --          fShiftKey := TRUE; (* dragging keeps selecting the rectangle *)
10 3014 -3      END;
10 3015 -2      END;
10 3016 -1      END;
10 3017 --      (* keep track of the old cell selection *)
10 3018 --      fOldCell := aCell;
10 3019 --      fNewMouseDown := FALSE;
10 3020 -0 A  END; (* TCellSelectCommand.TrackFeedback *)
10 3021 --
10 3022 --
10 3023 --      {*****}
10 3024 --      (* TCellSelectCommand.TrackMouse *)
10 3025 --      (* Track the mouse when the mouse button comes up. *)
10 3026 --      {*****}
10 3027 --      {$S GVRes}
10 3028 -- A  FUNCTION TCellSelectCommand.TrackMouse (aTrackPhase: TrackPhase;
10 3029 --      VAR anchorPoint, previousPoint, nextPoint: VPoint;
10 3030 --      mouseDidMove: BOOLEAN): TCommand; OVERRIDE;
10 3031 0- A  BEGIN
10 3032 --      WITH fGridView DO
10 3033 1-      BEGIN
10 3034 --          IF (aTrackPhase = TrackRelease) THEN
10 3035 2-      BEGIN
10 3036 --              (* set the new selection *)
10 3037 --              IF CanSelectCell (fOldCell) THEN
10 3038 3-      BEGIN
10 3039 --                  IF fSingleSelection THEN
10 3040 --                      SelectCell (fOldCell, kDontExtend, kDontHighlight, kSelect)
10 3041 --                  ELSE
10 3042 --                      SetSelection (fHLRegion, kDontExtend, kDontHighlight, kSelect);
10 3043 -3      END;
10 3044 --                      fNewMouseDown := TRUE;
10 3045 -2      END;
10 3046 -1      END; (* with *)
10 3047 --
10 3048 --      TrackMouse := SELF;
10 3049 -0 A  END; (* TCellSelectCommand.TrackMouse *)
10 3050 --
10 3051 --
10 3052 --      {*****}
10 3053 --      (* TCellSelectCommand.Fields *)
10 3054 --      (* For Debugging support of the inspector. *)
10 3055 --      {*****}
10 3056 --      {$IFC qDebug}
10 3057 --      {$S GVFields}
10 3058 -- A  PROCEDURE TCellSelectCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
10 3059 --      fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 3060 0- A  BEGIN
10 3061 --      DoToField ('TCellSelectCommand', NIL, bClass);
10 3062 --      DoToField ('fOldCell', @fOldCell, bInteger);
10 3063 --      DoToField ('fNewMouseDown', @fNewMouseDown, bBoolean);
10 3064 --      DoToField ('fKeepSelecting', @fKeepSelecting, bBoolean);
10 3065 --
10 3066 --      INHERITED Fields (DoToField);
10 3067 -0 A  END; (* TCellSelectCommand.Fields *)
10 3068 --      {$ENDC qDebug}

```

```

10 3069 --
10 3070 --
10 3071 -- {*****}
10 3072 -- { * TRowSelectCommand * }
10 3073 -- {*****}
10 3074 -- {*****}
10 3075 -- { * TRowSelectCommand.IRowSelectCommand * }
10 3076 -- { * Initialize the row select command object. * }
10 3077 -- {*****}
10 3078 -- {$S GVOpen}
10 3079 -- A PROCEDURE TRowSelectCommand.IRowSelectCommand (itsView: TGridView;
10 3080 -- aRow: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
10 3081 0- A BEGIN
10 3082 -- IGridSelectCommand (cRowSelect, itsView, theShiftKey, theCmdKey);
10 3083 -- fTheRow := aRow;
10 3084 -0 A END; { * TRowSelectCommand.IRowSelectCommand * }
10 3085 --
10 3086 --
10 3087 -- {*****}
10 3088 -- { * TRowSelectCommand.Fields * }
10 3089 -- { * For Debugging support of the inspector. * }
10 3090 -- {*****}
10 3091 -- {$IFC qDebug}
10 3092 -- {$S GVFields}
10 3093 -- A PROCEDURE TRowSelectCommand.Fields ( PROCEDURE DoToField (fieldName: Str255;
10 3094 -- fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 3095 0- A BEGIN
10 3096 -- DoToField ('TRowSelectCommand', NIL, bClass);
10 3097 -- DoToField ('fTheRow', @fTheRow, bInteger);
10 3098 --
10 3099 -- INHERITED Fields(DoToField);
10 3100 -0 A END; { * TRowSelectCommand.Fields * }
10 3101 -- {$ENDC qDebug}
10 3102 --
10 3103 --
10 3104 -- {*****}
10 3105 -- { * TColSelectCommand * }
10 3106 -- {*****}
10 3107 -- {*****}
10 3108 -- { * TColSelectCommand.IColumnSelectCommand * }
10 3109 -- { * Initialize the column select command object. * }
10 3110 -- {*****}
10 3111 -- {$S GVOpen}
10 3112 -- A PROCEDURE TColSelectCommand.IColSelectCommand (itsView: TGridView;
10 3113 -- aColumn: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
10 3114 0- A BEGIN
10 3115 -- IGridSelectCommand (cColumnSelect, itsView, theShiftKey, theCmdKey);
10 3116 -- fTheColumn := aColumn;
10 3117 -0 A END; { * TColSelectCommand.IColSelectCommand * }
10 3118 --
10 3119 --
10 3120 -- {*****}
10 3121 -- { * TColSelectCommand.Fields * }
10 3122 -- { * For Debugging support of the inspector. * }
10 3123 -- {*****}
10 3124 -- {$IFC qDebug}
10 3125 -- {$S GVFields}
10 3126 -- A PROCEDURE TColSelectCommand.Fields ( PROCEDURE DoToField (fieldName: Str255;
10 3127 -- fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 3128 0- A BEGIN
10 3129 -- DoToField ('TColSelectCommand', NIL, bClass);
10 3130 -- DoToField ('fTheColumn', @fTheColumn, bInteger);
10 3131 --
10 3132 -- INHERITED Fields(DoToField);
10 3133 -0 A END; { * TColSelectCommand.Fields * }
10 3134 -- {$ENDC qDebug}
10 3135 --
10 3136 --
10 3137 -- {*****}
10 3138 -- { * TVertexSelectCommand * }
10 3139 -- {*****}
10 3140 -- {*****}
10 3141 -- { * TVertexSelectCommand.IVertexSelectCommand * }
10 3142 -- { * Initialize the vertex select command object. * }
10 3143 -- { * The "vertex" is the place where rows and columns meet. * }
10 3144 -- {*****}
10 3145 -- {$S GVOpen}
10 3146 -- A PROCEDURE TVertexSelectCommand.IVertexSelectCommand (itsView: TGridView;
10 3147 -- aRow, aColumn: INTEGER; theShiftKey, theCmdKey: BOOLEAN);
10 3148 0- A BEGIN
10 3149 -- IGridSelectCommand (cVertexSelect, itsView, theShiftKey, theCmdKey);
10 3150 -- fTheRow := aRow;
10 3151 -- fTheColumn := aColumn;
10 3152 -0 A END; { * TVertexSelectCommand.IVertexSelectCommand * }
10 3153 --
10 3154 --
10 3155 -- {*****}
10 3156 -- { * TVertexSelectCommand.Fields * }
10 3157 -- { * For Debugging support of the inspector. * }
10 3158 -- {*****}
10 3159 -- {$IFC qDebug}
10 3160 -- {$S GVFields}
10 3161 -- A PROCEDURE TVertexSelectCommand.Fields ( PROCEDURE DoToField (fieldName: Str255;
10 3162 -- fieldAddr: Ptr; fieldType: INTEGER)); OVERRIDE;
10 3163 0- A BEGIN
10 3164 -- DoToField ('TVertexSelectCommand', NIL, bClass);
10 3165 -- DoToField ('fTheRow', @fTheRow, bInteger);
10 3166 -- DoToField ('fTheColumn', @fTheColumn, bInteger);
10 3167 --

```



```
10 3168 --      INHERITED Fields (DoToField);
10 3169 -0 A      END: (* TVertexSelectCommand.Fields *)
10 3170 --      {$ENDC qDebug}
10 3171 --
10 3172 --
10 3173 --      {*****}
10 3174 --      { * End Of File: UGridView.incl.p * }
10 3175 --      {*****}
9 782 --
9 783 --      END.
9 784 --
9 785 --      {*****}
9 786 --      { * End of File: UGridView.p * }
9 787 --      {*****}
```

```

11 1  --  {SP}
11 2  --  { UList.p }
11 3  --  { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
11 4  --
11 5  --  UNIT UList;
11 6  --
11 7  --  { Unit UList defines object type TList, a simple dynamic array whose elements
11 8  --    are references to TObjects }
11 9  --
11 10 --  INTERFACE
11 11 --
11 12 --  {-----+
11 13 --  | Uses |
11 14 --  +-----}
11 15 --  USES
11 16 --      {$LOAD MacIntf.LOAD}
11 17 --      MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
11 18 --      {$LOAD}
11 19 --      UMAUtil,
11 20 --      UFailure,
11 21 --      UMemory,
11 22 --      UObject;
11 23 --
11 24 --
11 25 --  {-----+
11 26 --  | Global constants |
11 27 --  +-----}
11 28 --  CONST
11 29 --      kDeletedElement = $FFFFFFF;
11 30 --
11 31 --      kALessThanB = -1; { Constants for TSortedList.Compare and Search. }
11 32 --      kAEqualB = 0;
11 33 --      kAGreaterThanB = 1;
11 34 --
11 35 --
11 36 --  {-----+
11 37 --  | Global types |
11 38 --  +-----}
11 39 --  TYPE
11 40 --      {RW means OK for other objects to Read and Write this field}
11 41 --      {Rx means OK for other objects to Read, but NOT TO WRITE this field}
11 42 --      {xx means other objects should NEITHER READ NOR WRITE this field}
11 43 --
11 44 --      {A dynamic list of TObjects}
11 45 --
11 46 --  {-----+
11 47 --  | TList |
11 48 --  +-----}
11 49 --      TList = OBJECT(TObject) {It IS permissible for descendants to add
11 50 --                             more named fields}
11 51 --          fSize: INTEGER; {Rx}{number of elements in the list, from 0
11 52 --                           to the limit of memory}
11 53 --          fFirstOffset: LONGINT; {xx}{= SIZEOF(SELF) = no. of bytes of named
11 54 --                           fields before first element}
11 55 --          fDeletions: INTEGER; {Rx}{The number of deleted elements in the
11 56 --                           list. These have value
11 57 --                           kDeletedElement; fSize always
11 58 --                           reflects # of real elements.}
11 59 --          fEachLevel: INTEGER; {Rx}{Number of Each calls in progress.}
11 60 --      {$IFC qDebug}
11 61 --          fEltClass: String8; {xx}{if <> '', then the name of the class
11 62 --                           of the elements of the list}
11 63 --          fObjPtr: LONGINT; {xx}{internal}
11 64 --      {$ENDC}
11 65 --
11 66 --      {Creation/Destruction Methods}
11 67 --
11 68 --      PROCEDURE TList.IList;
11 69 --          {Initialize a new list with no elements, i.e., fSize = 0. Always
11 70 --          call it once before calling any other method. Never call it
11 71 --          twice.}
11 72 --
11 73 --      PROCEDURE TList.FreeList;
11 74 --          {Frees each object in the list, then frees the list.}
11 75 --
11 76 --      {Utility Methods}
11 77 --
11 78 --      FUNCTION TList.GetSize: INTEGER;
11 79 --          {Returns the number of elements in the list.}
11 80 --
11 81 --      FUNCTION TList.At(index: INTEGER): TObject;
11 82 --          {Return the index'th element of the list. It is typical for the
11 83 --          caller to coerce the result into a descendant of TObject.
11 84 --          All lists are indexed from 1. Range check only if the
11 85 --          compile-flag qRangeCheck is TRUE.
11 86 --          The static-array equivalent of:
11 87 --          object := objList.At(index);
11 88 --          is:
11 89 --          object := objArray[index];}
11 90 --
11 91 --      FUNCTION TList.GetItemNumber(item: TObject): INTEGER;
11 92 --          {Return the index of the first occurrence of item in the list,
11 93 --          or fSize + 1 if item is not in the list.}
11 94 --
11 95 --      FUNCTION TList.First: TObject;
11 96 --          {Return the first element of the list. It is typical for the
11 97 --          caller to coerce the result. Returns NIL if the size is <= 0.}
11 98 --

```

```

11 99 --      The static-array equivalent of:
11 100 --      object := objList.First;
11 101 --      is:
11 102 --      object := objArray[1];
11 103 --
11 104 --      FUNCTION TList.Last: TObject;
11 105 --      (Return the last element of the list. It is typical for the
11 106 --      caller to coerce the result. Returns NIL if the size is <= 0.
11 107 --
11 108 --      The static-array equivalent of:
11 109 --      object := objList.Last;
11 110 --      is:
11 111 --      object := objArray[number of elements];
11 112 --
11 113 --      (Iterator Methods)
11 114 --
11 115 --      PROCEDURE TList.Each(PROCEDURE DoToItem(item: TObject));
11 116 --      (Call DoToItem once for each element of the list, in order. The
11 117 --      actual parameter is typically a procedure whose argument is a
11 118 --      descendant of TObject. If DoToItem calls InsertLast, the newly
11 119 --      added element will NOT be enumerated. If TestItem calls AtPut,
11 120 --      InsertBefore, InsertFirst, Delete, or DeleteAll, misbehavior
11 121 --      will ensue. If DoToItem calls Delete, the item is replaced by
11 122 --      the constant kDeletedElement. fSize is decremented and
11 123 --      fDeletions is incremented. At the end of all calls to Each,
11 124 --      TList.RemoveDeletions will be called.
11 125 --
11 126 --      The static-array equivalent of:
11 127 --      objList.Each(Proc)
11 128 --      is:
11 129 --      FOR index := 1 TO fSize DO
11 130 --      Proc(objArray[index]);
11 131 --
11 132 --      FUNCTION TList.FirstThat(FUNCTION TestItem(item: TObject): BOOLEAN):
11 133 --      TObject;
11 134 --      (Call TestItem once for each element of the list, in order, until
11 135 --      TestItem returns TRUE. Return the element that satisfied the
11 136 --      test. If none satisfied the test, return NIL. The actual
11 137 --      parameter is typically a procedure whose argument is a
11 138 --      descendant of TObject. It is typical for the caller to coerce
11 139 --      the result into that descendant of TObject. If TestItem calls
11 140 --      InsertLast, the newly added element will NOT be enumerated.
11 141 --      If TestItem calls AtPut, InsertBefore, InsertFirst, Delete, or
11 142 --      DeleteAll, misbehavior will ensue.
11 143 --
11 144 --      The static-array equivalent of:
11 145 --      object := objList.FirstThat(Func)
11 146 --      is:
11 147 --      object := NIL;
11 148 --      FOR index := 1 TO fSize DO IF Func(objArray[index]) THEN
11 149 --      BEGIN
11 150 --      object := objArray[index];
11 151 --      LEAVE;
11 152 --      END)
11 153 --
11 154 --      FUNCTION TList.LastThat(FUNCTION TestItem(item: TObject): BOOLEAN):
11 155 --      TObject;
11 156 --      (Call TestItem once for each element of the list, starting with
11 157 --      the last item in the list and working toward the first item,
11 158 --      until TestItem returns TRUE. Return the element that satisfied
11 159 --      the test. If none satisfied the test, return NIL. The actual
11 160 --      parameter is typically a procedure whose argument is a
11 161 --      descendant of TObject. It is typical for the caller to coerce
11 162 --      the result into that descendant of TObject. If TestItem calls
11 163 --      InsertLast, the newly added element will NOT be enumerated.
11 164 --      If TestItem calls AtPut, InsertBefore, InsertFirst, Delete, or
11 165 --      DeleteAll, misbehavior will ensue.
11 166 --
11 167 --      The static-array equivalent of:
11 168 --      object := objList.LastThat(Func)
11 169 --      is:
11 170 --      object := NIL;
11 171 --      FOR index := fSize DOWNTO 1 DO IF Func(objArray[index]) THEN
11 172 --      BEGIN
11 173 --      object := objArray[index];
11 174 --      LEAVE;
11 175 --      END)
11 176 --
11 177 --      (Item Insertion Methods)
11 178 --
11 179 --      PROCEDURE TList.AtPut(index: INTEGER; newItem: TObject);
11 180 --      (Replace the index'th item of the list. Does not free the old
11 181 --      item. If you do this from within an Each the results will be
11 182 --      unpredictable.)
11 183 --
11 184 --      PROCEDURE TList.InsertBefore(index: INTEGER; item: TObject);
11 185 --      (Insert a reference to an item before the indicated item. The
11 186 --      index of the new element will be index.
11 187 --
11 188 --      If index = 1 this inserts at the start of the list.
11 189 --      If index = fSize+1 this inserts at the end of the list.
11 190 --
11 191 --      Signals Failure if unable to increase the size of the list.)
11 192 --
11 193 --
11 194 --      PROCEDURE TList.InsertFirst(item: TObject);
11 195 --      (Insert a reference to item at the front of the list. If the
11 196 --      compile-flag qDebug is TRUE & SetEltType was called, verify
11 197 --      item's type. Increase fSize by 1. The index of the new

```

```

11 198 --          element will be 1.  Signals Failure if unable to increase the
11 199 --          size of the list.)
11 200 --
11 201 --
11 202 --      PROCEDURE TList.InsertLast(item: TObject);
11 203 --          {Insert a reference to item at the back of the list.  If the
11 204 --           compile-flag qDebug is TRUE & SetElementType was called, verify
11 205 --           item's type.  Increase fSize by 1.  The index of the new
11 206 --           element will be fSize.
11 207 --
11 208 --           Signals Failure if unable to increase the size of the list.}
11 209 --
11 210 --      {Item Deletion Methods}
11 211 --
11 212 --      PROCEDURE TList.Delete(item: TObject);
11 213 --          {Delete the first reference to item from the list, but do not free
11 214 --           item.  If item does not occur, do nothing.  If item does occur,
11 215 --           reduce fSize by 1.}
11 216 --
11 217 --      PROCEDURE TList.DeleteAll;
11 218 --          {Delete every element from the list, but do not free any objects.
11 219 --           Leave fSize at 0.}
11 220 --
11 221 --      PROCEDURE TList.FreeAll;
11 222 --          {Frees each element in the list.  Leave fSize at 0.}
11 223 --
11 224 --      {Internal Methods}
11 225 --
11 226 --      FUNCTION TList.ComputeAddress (index: INTEGER): LONGINT;
11 227 --          {Returns the memory address of the index-th element in the list.
11 228 --           Used internally.}
11 229 --
11 230 --      FUNCTION TList.MaxAddress: LONGINT;
11 231 --          {Returns the memory address after the last item in the list.
11 232 --           Used internally.}
11 233 --
11 234 --      PROCEDURE TList.RemoveDeletions;
11 235 --          {Remove the deleted elements from the list.}
11 236 --
11 237 --      {Debugging Methods}
11 238 --
11 239 --      {$IFC qDebug}
11 240 --          PROCEDURE TList.SetElementType(toClass: Str255);
11 241 --              {Call this once and pass the kind of objects you intend to insert
11 242 --               into the list.  The TList insert methods will do a coercion to
11 243 --               the type you specify, to ensure that the list contains only
11 244 --               elements of the right type.}
11 245 --
11 246 --          PROCEDURE TList.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
11 247 --          PROCEDURE TList.Fields (PROCEDURE DoToField (fieldName: Str255;
11 248 --              fieldAddr: Ptr;
11 249 --              fieldType: INTEGER)); OVERRIDE;
11 250 --      {$ENDC}
11 251 --
11 252 --      END (TList);
11 253 --
11 254 --
11 255 --      {-----+
11 256 --      | TSortedList |
11 257 --      +-----+}
11 258 --      TSortedList = OBJECT(TList)
11 259 --
11 260 --      FUNCTION TSortedList.GetItemNumber(item: TObject): INTEGER; OVERRIDE;
11 261 --          { If item is in the list, GetItemNumber returns an index number to an occurrence
11 262 --           of item.  If the item is in the list more than once, then the index returned
11 263 --           is not guaranteed to be the first occurrence of the item because a binary
11 264 --           search is used.
11 265 --
11 266 --           If the item is not in the list, then GetItemNumber returns the index at
11 267 --           which the item should be inserted into the list.  This GetItemNumber
11 268 --           does not tell you whether the item is in fact in the list.  This can be
11 269 --           determined by calling the Search method or by calling the At method on
11 270 --           the result of GetItemNumber.}
11 271 --
11 272 --      PROCEDURE TSortedList.Insert (item: TObject);
11 273 --          { Inserts the given item into the list in sorted order by using a binary
11 274 --           search.  The Compare method is used to determine the items location in the
11 275 --           list.}
11 276 --
11 277 --      FUNCTION TSortedList.Compare (item1, item2: TObject): INTEGER;
11 278 --          { This method compares to items in the list.  It must be overridden.
11 279 --           A negative result indicates that item1 < item2.  A result of zero indicates
11 280 --           that item1 = item2.  A positive result indicates that item1 > item2.
11 281 --           You may wish to use the constants kALessThanB, kAEqualB, kAGreaterThanB. }
11 282 --
11 283 --      FUNCTION TSortedList.Search (FUNCTION TestItem(anItem: TObject): INTEGER): TObject;
11 284 --          { This method performs a binary search on the list in which you provide
11 285 --           the function that tests items in the list.  This is useful for cases
11 286 --           in which you are not comparing objects in the list with another object,
11 287 --           as does Compare.  For example, suppose each object has an fTitle field
11 288 --           and the objects are inserted into the list in order of fTitle.  You
11 289 --           can use the search method to look for an object whose fTitle is equal
11 290 --           to any string.
11 291 --
11 292 --           A negative TestItem result indicates that anItem < your search criteria,
11 293 --           a result of zero indicates the item has been found and Search returns that
11 294 --           item.  A positive TestItem result indicates that anItem > your search
11 295 --           criteria. }
11 296 --

```

```
11 297 --          END (TSortedList);
11 298 --
11 299 --
11 300 --      {-----+
11 301 --      | Global Routines |
11 302 --      +-----}
11 303 --      FUNCTION NewList: TList;
11 304 --          (A convenience function. Create a TList, initialize it, and
11 305 --           return a reference to it.
11 306 --
11 307 --           Signals Failure if it cannot allocate the object.)
11 308 --
11 309 --
11 310 --      IMPLEMENTATION
11 311 --
11 312 --      {$I UList.incl.p}
```

```

12 1 --      {$P}
12 2 --      { UList.incl.p }
12 3 --      { Copyright © 1986-1988 by Apple Computer, Inc. All rights reserved. }
12 4 --      (*
12 5 --      Need more testing of Deletions & At calls during Each; multiple Each's
12 6 --      *)
12 7 --
12 8 --      {-----+
12 9 --      | Local constants |
12 10 --      +-----}
12 11 --      CONST
12 12 --      kElemSize = SIZEOF(Handle);      ( Each element is really a handle )
12 13 --
12 14 --      {-----+
12 15 --      | Local types |
12 16 --      +-----}
12 17 --      TYPE
12 18 --      PObj =      ^TObject;
12 19 --
12 20 --      {-----+
12 21 --      | External declarations |
12 22 --      +-----}
12 23 --      {$#+ }
12 24 -- A FUNCTION  _InObj(ordObject, jumpTablePtr: LONGINT): BOOLEAN; EXTERNAL;
12 25 --      {$%-}
12 26 --
12 27 --
12 28 --      (*****
12 29 --      (*      G l o b a l   R o u t i n e s
12 30 --      (*****
12 31 --      {$S ListRes)
12 32 --      {-----+
12 33 --      | NewList |
12 34 --      +-----}
12 35 -- A FUNCTION NewList: TList;
12 36 --      ( A convenience function.
12 37 --      Create a TList, initialize it, and return a reference to it. )
12 38 --      VAR
12 39 --      list: TList;
12 40 -- A BEGIN
12 41 --      New(list);
12 42 --      FailNil(list);
12 43 --
12 44 --      list.IList;
12 45 --      NewList := list;
12 46 -- A END;
12 47 --
12 48 --
12 49 --      (*****
12 50 --      (*      T L I s t
12 51 --      (*****
12 52 --      {$S ListRes)
12 53 --      {-----+
12 54 --      | IList |
12 55 --      +-----}
12 56 -- A PROCEDURE TList.IList;
12 57 --      ( Initialize a new list with no elements, i.e., fSize = 0 )
12 58 -- A BEGIN
12 59 --      fSize := 0;
12 60 --      ( We store the items in the list after the last declared field of the object.
12 61 --      fFirstOffset is the offset into the object where the first item is stored. )
12 62 --      fFirstOffset := GetHandleSize(Handle(SELF));
12 63 --      fEachLevel := 0;
12 64 --      fDeletions := 0;
12 65 --      ($IFC qDebug)
12 66 --      fEltClass := '';
12 67 --      fObjPtr := 0;
12 68 --      ($ENDC)
12 69 -- A END;
12 70 --
12 71 --
12 72 --      {$S ListRes)
12 73 --      {-----+
12 74 --      | At |
12 75 --      +-----}
12 76 -- A FUNCTION TList.At(index: INTEGER): TObject;
12 77 --      ( Return the index'th element of the list.
12 78 --      Range check only if the compile-flag qRangeCheck is TRUE. )
12 79 --      VAR
12 80 --      p:      LONGINT;
12 81 --      item:    LONGINT;
12 82 --      x:      INTEGER;
12 83 -- A BEGIN
12 84 --      ($IFC qRangeCheck)
12 85 --      IF (index < 1) OR (index > fSize) THEN
12 86 --      BEGIN
12 87 --      ($IFC qDebug)
12 88 --      Writeln('fSize = ', fSize:1, ' index = ', index:1);
12 89 --      ProgramBreak('Range Check in TList.At');
12 90 --      ($ENDC)
12 91 --      END;
12 92 --      ($ENDC)
12 93 --
12 94 --      At := PObj(ComputeAddress(index))^;
12 95 -- A END;
12 96 --
12 97 --
12 98 --      {$S ListRes)

```

```

12 99 -- {-----+
12 100 -- | AtPut |
12 101 -- +-----+
12 102 -- A PROCEDURE TList.AtPut(index: INTEGER; newItem: TObject);
12 103 -- { Replace the index'th element of the list.
12 104 -- Range check only if the compile-flag qRangeCheck is TRUE. }
12 105 -- VAR
12 106 -- p: POBJ;
12 107 -- x: INTEGER;
12 108 -- item: LONGINT;
12 109 0- A BEGIN
12 110 -- {SIFC qRangeCheck}
12 111 -- IF (index < 1) OR (index > fSize) THEN
12 112 1- BEGIN
12 113 -- {SIFC qDebug}
12 114 -- Writeln('fSize = ', fSize:1, ' index = ', index:1);
12 115 -- ProgramBreak('Range Check in TList.AtPut');
12 116 -- {$ENDC}
12 117 -1 END;
12 118 -- {$ENDC}
12 119 --
12 120 -- {%%+}
12 121 -- {SIFC qDebug}
12 122 -- IF fObjPtr > 0 THEN
12 123 -- IF NOT %InObj(ORD(newItem), fObjPtr) THEN
12 124 1- BEGIN
12 125 -- WritelnPtr('newItem', newItem);
12 126 -- Writeln;
12 127 -- ProgramBreak('In TList.AtPut: inserting invalid object type');
12 128 -1 END;
12 129 -- {$ENDC}
12 130 -- {%-}
12 131 --
12 132 -- p := POBJ(ComputeAddress(index));
12 133 -- p^ := newItem;
12 134 -0 A END;
12 135 --
12 136 --
12 137 -- { ELEMENT-ADDRESS-COMPUTATION used in many TList methods:
12 138 -- Ord(Handle(SELF)^) = address of the class pointer within the heap block
12 139 -- fFirstOffset = no. of bytes of fixed fields - offset from class pointer to first element
12 140 -- (kElemSize * index) - kElemSize =
12 141 -- kElemSize * (index - 1) = no. of bytes in elements preceding the index'th element
12 142 -- NOTES:
12 143 -- Direct heap pointers are not preserved across procedure calls that might compact the heap.
12 144 -- The ROM routine Munger can do fast inserts and deletes. }
12 145 --
12 146 --
12 147 -- {$S ListRes}
12 148 -- {-----+
12 149 -- | ComputeAddress |
12 150 -- +-----+
12 151 -- A FUNCTION TList.ComputeAddress (index: INTEGER): LONGINT;
12 152 -- VAR
12 153 -- p: LONGINT;
12 154 -- x: INTEGER;
12 155 -- item: LONGINT;
12 156 0- A BEGIN
12 157 -- p := Ord(Handle(SELF)^) + fFirstOffset; { Address of first physical item }
12 158 --
12 159 -- IF fDeletions = 0 THEN { If no deletions, index to proper element }
12 160 -- ComputeAddress := p + BSL(index - 1, 2) { p + ((index - 1) * 4) }
12 161 -- ELSE { Ignore deleted elements }
12 162 1- BEGIN
12 163 -- x := index;
12 164 -- p := p - kElemSize; { we increment first in loop }
12 165 --
12 166 -- WHILE x > 0 DO { Skip elements until item is found }
12 167 2- BEGIN
12 168 -- p := p + kElemSize;
12 169 -- item := PLongInt(p)^;
12 170 -- IF item <> kDeletedElement THEN
12 171 -- x := x - 1;
12 172 -2 END;
12 173 --
12 174 -- ComputeAddress := p;
12 175 -1 END
12 176 -0 A END;
12 177 --
12 178 --
12 179 -- {$S ListRes}
12 180 -- {-----+
12 181 -- | Delete |
12 182 -- +-----+
12 183 -- A PROCEDURE TList.Delete(item: TObject);
12 184 -- { Delete the first reference to item in the list, but do not free item.
12 185 -- If item does not occur, do nothing. }
12 186 -- VAR
12 187 -- p: LONGINT;
12 188 -- maxP: LONGINT;
12 189 -- offset: LONGINT;
12 190 0- A BEGIN
12 191 -- { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 192 -- p := ComputeAddress(1); { Address of first element }
12 193 -- maxP := MaxAddress; { Address past last element }
12 194 --
12 195 -- { Skip elements until item is found }
12 196 -- WHILE (p < maxP) & (POBJ(p)^ <> item) DO
12 197 -- p := p + kElemSize;

```

```

12 198 --
12 199 --      IF p < maxP THEN { Item was found }
12 200 1-      BEGIN
12 201 --          IF fEachLevel > 0 THEN
12 202 2-          BEGIN
12 203 --              fDeletions := fDeletions + 1;
12 204 --              PLongInt(p)^ := kDeletedElement;
12 205 -2          END
12 206 --          ELSE
12 207 --              offset := Munger(Handle(SELF), p - Ord(Handle(SELF)^), NIL,
12 208 --                  kElemSize, Ptr(@item), 0);
12 209 --
12 210 --              fSize := fSize - 1;
12 211 -1          END;
12 212 -0 A      END;
12 213 --
12 214 --
12 215 --      {$S ListRes}
12 216 --      {-----+
12 217 --      | DeleteAll |
12 218 --      +-----}
12 219 -- A      PROCEDURE TList.DeleteAll;
12 220 --      { Delete every element from the list, but do not free any objects. }
12 221 0- A      BEGIN
12 222 --          {$IFC qDebug}
12 223 --          IF fEachLevel > 0 THEN
12 224 --              ProgramBreak('In TList.DeleteAll: currently doing an Each call! (fEachLevel > 0)');
12 225 --          {$ENDC}
12 226 --
12 227 --          { Delete the elements from the heap block }
12 228 --          SetHandleSize(Handle(SELF), fFirstOffset);
12 229 --          fSize := 0;
12 230 -0 A      END;
12 231 --
12 232 --
12 233 --      {$S ListRes}
12 234 --      {-----+
12 235 --      | Each |
12 236 --      +-----}
12 237 -- A      PROCEDURE TList.Each(PROCEDURE DoToItem(item: TObject));
12 238 --      { Call DoToItem once for each element of the list, in order. }
12 239 --      VAR
12 240 --          index:    INTEGER;
12 241 --          offset:   LONGINT;
12 242 --          item:     LONGINT;
12 243 --          fi:       FailInfo;
12 244 --
12 245 --      {-----+
12 246 --      | FinishEach |
12 247 --      +-----}
12 248 -- B      PROCEDURE FinishEach;
12 249 0- B      BEGIN
12 250 --          fEachLevel := fEachLevel - 1;
12 251 --          IF fEachLevel = 0 THEN
12 252 --              IF fDeletions > 0 THEN
12 253 --                  RemoveDeletions;
12 254 -0 B      END;
12 255 --
12 256 --      {-----+
12 257 --      | Hd1Each |
12 258 --      +-----}
12 259 -- B      PROCEDURE Hd1Each(error: OSerr; message: LONGINT);
12 260 0- B      BEGIN
12 261 --          FinishEach;
12 262 -0 B      END;
12 263 --
12 264 0- A      BEGIN
12 265 --          IF fSize > 0 THEN
12 266 1-          BEGIN
12 267 --              fEachLevel := fEachLevel + 1;
12 268 --
12 269 --              CatchFailures(fi, Hd1Each);
12 270 --
12 271 --              offset := fFirstOffset;
12 272 --
12 273 --              { Note that upper bound of the loop is the value of fSize at the
12 274 --              time the loop is begun. Changing fSize in the loop has no
12 275 --              effect on the number of times the loop is executed. }
12 276 --
12 277 --              FOR index := 1 TO fSize DO
12 278 2-              BEGIN
12 279 --                  { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 280 --                  { must recompute every time because of heap compaction }
12 281 --                  item := PLongInt(Ord(Handle(SELF)^) + offset)^;
12 282 --                  IF item <> kDeletedElement THEN
12 283 --                      DoToItem(TObject(item));
12 284 --                      offset := offset + kElemSize;
12 285 -2                  END;
12 286 --
12 287 --                  Success(fi);
12 288 --
12 289 --                  FinishEach;
12 290 -1              END;
12 291 -0 A      END;
12 292 --
12 293 --
12 294 --      {$S ListRes}
12 295 --      {-----+
12 296 --      | First |

```



```

12 297 -- +-----}
12 298 -- A FUNCTION TList.First: TObject;
12 299 -- { Return the first element of the list.
12 300 -- { If the compile-flag qDebug is TRUE & SetEltType was called, verify item's type. }
12 301 0- A BEGIN
12 302 -- IF fSize <= 0 THEN
12 303 --     First := NIL
12 304 -- ELSE
12 305 --     First := At(1);
12 306 -0 A END;
12 307 --
12 308 --
12 309 -- {$S ListRes}
12 310 -- {-----+
12 311 -- | FirstThat |
12 312 -- +-----}
12 313 -- A FUNCTION TList.FirstThat(FUNCTION TestItem(item: TObject): BOOLEAN): TObject;
12 314 -- VAR
12 315 --     index:    INTEGER;
12 316 --     offset:   LONGINT;
12 317 --     anObject: TObject;
12 318 --
12 319 0- A BEGIN
12 320 --     offset := fFirstOffset;
12 321 --
12 322 --     FOR index := 1 TO fSize DO
12 323 1- BEGIN
12 324 --         { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 325 --         { must recompute every time because of heap compaction }
12 326 --         anObject := PObj(Ord(Handle(SELF)^) + offset)^;
12 327 --         IF TestItem(anObject) THEN { Found }
12 328 2- BEGIN
12 329 --             FirstThat := anObject;
12 330 --             EXIT(FirstThat);
12 331 -2 END;
12 332 --             offset := offset + kElemSize;
12 333 -1 END;
12 334 --         FirstThat := NIL;
12 335 -0 A END;
12 336 --
12 337 --
12 338 -- {$S ListRes}
12 339 -- {-----+
12 340 -- | FreeAll |
12 341 -- +-----}
12 342 -- A PROCEDURE TList.FreeAll;
12 343 --
12 344 -- B PROCEDURE FreeObject (theObject: TObject);
12 345 --
12 346 0- B BEGIN
12 347 --     theObject.Free;
12 348 -0 B END;
12 349 --
12 350 0- A BEGIN
12 351 --     Each(FreeObject);
12 352 --     DeleteAll;
12 353 -0 A END;
12 354 --
12 355 --
12 356 -- {$S ListRes}
12 357 -- {-----+
12 358 -- | FreeList |
12 359 -- +-----}
12 360 -- A PROCEDURE TList.FreeList;
12 361 --
12 362 -- B PROCEDURE FreeObject (theObject: TObject);
12 363 --
12 364 0- B BEGIN
12 365 --     theObject.Free;
12 366 -0 B END;
12 367 --
12 368 0- A BEGIN
12 369 --     Each(FreeObject);
12 370 --     Free;
12 371 -0 A END;
12 372 --
12 373 --
12 374 -- {$S ListRes}
12 375 -- {-----+
12 376 -- | GetItemNumber |
12 377 -- +-----}
12 378 -- A FUNCTION TList.GetItemNumber(item: TObject): INTEGER;
12 379 -- { Find the first reference to item in the list.
12 380 -- { If item does not occur, return fSize + 1. }
12 381 -- VAR
12 382 --     p:      LONGINT;
12 383 --     i:      LONGINT;
12 384 --     maxP:   LONGINT;
12 385 --     offset: LONGINT;
12 386 0- A BEGIN
12 387 --     { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 388 --     p := ComputeAddress(1); { Address of first element }
12 389 --     i := 1;
12 390 --     maxP := MaxAddress; { Address past last element }
12 391 --
12 392 --     { Skip elements until item is found }
12 393 1- WHILE (p < maxP) & (PObj(p)^ <> item) DO BEGIN
12 394 --         p := p + kElemSize;
12 395 --         IF p <> kDeletedElement THEN

```

```

12 396 --      i := i + 1;
12 397 -1      END;
12 398 --
12 399 --      GetItemNumber := i;
12 400 -0 A  END;
12 401 --
12 402 --      {$S ListRes}
12 403 --      {-----+
12 404 --      |   GetSize
12 405 --      +-----}
12 406 -- A  FUNCTION TList.GetSize: INTEGER;
12 407 --      { Returns the number of elements in the list. }
12 408 --
12 409 0- A  BEGIN
12 410 --      GetSize := fSize;
12 411 -0 A  END;
12 412 --
12 413 --
12 414 --      {$S ListRes}
12 415 --      {-----+
12 416 --      |   InsertBefore
12 417 --      +-----}
12 418 -- A  PROCEDURE TList.InsertBefore(index: INTEGER; item: TObject);
12 419 --      { Insert a reference to item at the indicated index.
12 420 --        If the compile-flag qDebug is TRUE & SetElType was called, verify item's type. }
12 421 --
12 422 --      VAR
12 423 --          oldSize:   Size;
12 424 --          offset:    LONGINT;
12 425 --          permFlag:  BOOLEAN;
12 426 --
12 427 0- A  BEGIN
12 428 --      {$IFC qDebug}
12 429 --          IF fEachLevel > 0 THEN
12 430 --              ProgramBreak('In TList.InsertBefore: currently doing an Each call! (fEachLevel > 0)');
12 431 --
12 432 --          IF (index < 1) OR (index > fSize+1) THEN
12 433 1-      BEGIN
12 434 --              Writeln('fSize = ', fSize:1, ' index = ', index:1);
12 435 --              ProgramBreak('Range Check in TList.InsertBefore');
12 436 -1      END;
12 437 --
12 438 --      {$%+}
12 439 --          IF fObjPtr > 0 THEN
12 440 --              IF NOT % InObj(ORD(item), fObjPtr) THEN
12 441 1-      BEGIN
12 442 --              Writeln('item', item);
12 443 --              Writeln;
12 444 --              ProgramBreak('In TList.InsertBefore: inserting invalid object type');
12 445 -1      END;
12 446 --      {$%-}
12 447 --      {$ENDC qDebug}
12 448 --
12 449 --      { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 450 --      oldSize := GetHandleSize(Handle(SELF));
12 451 --
12 452 --      permFlag := PermAllocation(TRUE);
12 453 --
12 454 --      { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 455 --      offset := Munger(Handle(SELF),
12 456 --          fFirstOffset + BSL(index-1, 2),    { index-1 * 4 }
12 457 --          NIL, 0, Ptr(@item), kElemSize);
12 458 --
12 459 --      permFlag := PermAllocation(permFlag);
12 460 --
12 461 --      IF GetHandleSize(Handle(SELF)) <= oldSize THEN
12 462 --          Failure(memFullErr, 0);
12 463 --
12 464 --      fSize := fSize + 1;
12 465 -0 A  END;
12 466 --
12 467 --
12 468 --      {$S ListRes}
12 469 --      {-----+
12 470 --      |   InsertFirst
12 471 --      +-----}
12 472 -- A  PROCEDURE TList.InsertFirst(item: TObject);
12 473 --      { Insert a reference to item at the front of the list.
12 474 --        If the compile-flag qDebug is TRUE & SetElType was called, verify item's type. }
12 475 0- A  BEGIN
12 476 --      InsertBefore(1, item);
12 477 -0 A  END;
12 478 --
12 479 --
12 480 --      {-----+
12 481 --      |   InsertLast
12 482 --      +-----}
12 483 -- A  PROCEDURE TList.InsertLast(item: TObject);
12 484 --      { Insert a reference to item at the back of the list.
12 485 --        If qDebug is TRUE & SetElType was called, check that item's type is correct. }
12 486 0- A  BEGIN
12 487 --      InsertBefore(fSize+1, item);
12 488 -0 A  END;
12 489 --
12 490 --
12 491 --      {$S ListRes}
12 492 --      {-----+
12 493 --      |   Last
12 494 --      +-----}

```

```

12 495 -- A FUNCTION TList.Last: TObject;
12 496 --     { Return the last element of the list.
12 497 --     If the compile-flag qDebug is TRUE & SetEltType was called, verify item's type. }
12 498 0- A BEGIN
12 499 --     IF fSize <= 0 THEN
12 500 --         Last := NIL
12 501 --     ELSE
12 502 --         Last := At(fSize);
12 503 -0 A END;
12 504 --
12 505 --
12 506 --     {$S ListRes}
12 507 --     {-----+
12 508 --     | LastThat
12 509 --     +-----}
12 510 -- A FUNCTION TList.LastThat(FUNCTION TestItem(item: TObject): BOOLEAN): TObject;
12 511 -- VAR
12 512 --     index: INTEGER;
12 513 --     offset: LONGINT;
12 514 --     anObject: TObject;
12 515 --
12 516 0- A BEGIN
12 517 --     offset := fFirstOffset + (fSize - 1) * kElemSize;
12 518 --
12 519 --     FOR index := 1 TO fSize DO
12 520 1- BEGIN
12 521 --         { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 522 --         { must recompute every time because of heap compaction }
12 523 --         anObject := POBJ(Ord(Handle(SELF)^) + offset)^;
12 524 --         IF TestItem(anObject) THEN { Found }
12 525 2- BEGIN
12 526 --             LastThat := anObject;
12 527 --             EXIT(LastThat);
12 528 -2 END;
12 529 --         offset := offset - kElemSize;
12 530 -1 END;
12 531 --     LastThat := NIL;
12 532 -0 A END;
12 533 --
12 534 --
12 535 --     { Return address after last element of list }
12 536 --     {$S ListRes}
12 537 --     {-----+
12 538 --     | MaxAddress
12 539 --     +-----}
12 540 -- A FUNCTION TList.MaxAddress: LONGINT;
12 541 0- A BEGIN
12 542 --     MaxAddress := Ord(Handle(SELF)^) + GetHandleSize(Handle(SELF));
12 543 -0 A END;
12 544 --
12 545 --
12 546 --     {$S ListRes}
12 547 --     {-----+
12 548 --     | RemoveDeletions
12 549 --     +-----}
12 550 -- A PROCEDURE TList.RemoveDeletions;
12 551 -- VAR
12 552 --     srcP: LONGINT;
12 553 --     dstP: LONGINT;
12 554 --     maxP: LONGINT;
12 555 --     elt: LONGINT;
12 556 0- A BEGIN
12 557 --     { See ELEMENT-ADDRESS-COMPUTATION comment above }
12 558 --     { Can't use ComputeAddress because it would bypass deleted elements. }
12 559 --     srcP := Ord(Handle(SELF)^) + fFirstOffset; { Address of first physical item }
12 560 --     dstP := srcP;
12 561 --     maxP := MaxAddress; { Address past last element }
12 562 --
12 563 --     { Compact the non deleted elements to start of list. }
12 564 --     WHILE srcP < maxP DO
12 565 1- BEGIN
12 566 --         elt := PLongInt(srcP)^;
12 567 --         IF elt <> kDeletedElement THEN
12 568 2- BEGIN
12 569 --             PLongInt(dstP)^ := elt;
12 570 --             dstP := dstP + kElemSize;
12 571 -2 END;
12 572 --         srcP := srcP + kElemSize;
12 573 -1 END;
12 574 --
12 575 --     fDeletions := 0;
12 576 --     SetHandleSize(Handle(SELF), fFirstOffset + BSL(fSize, 2)); {fSize * 4}
12 577 -0 A END;
12 578 --
12 579 --
12 580 --     {%-}
12 581 --     {$IFC qDebug}
12 582 --     {$S ListDebug}
12 583 --     {-----+
12 584 --     | SetEltType
12 585 --     +-----}
12 586 -- A PROCEDURE TList.SetEltType(toClass: Str255);
12 587 --     { Initialize fEltClass to toClass. fEltClass is a STRING[8] because identifiers
12 588 --     are unique to 8 chars }
12 589 -- VAR
12 590 --     s: String8;
12 591 0- A BEGIN
12 592 --     { Truncate to 8 characters, and force upper case }
12 593 --     IF Length(toClass) > 8 THEN

```

```

12 594 --      s := Copy(toClass, 1, 8)
12 595 --      ELSE
12 596 --      s := toClass;
12 597 --      UprString8(s);
12 598 --
12 599 --      { Forgive the caller for calling this twice only if the same type was specified both times }
12 600 --      IF fEltClass <> '' THEN
12 601 --      IF fEltClass <> s THEN
12 602 1-      BEGIN
12 603 --      Writeln('The list already contains ', fEltClass, ': trying to change to ', s);
12 604 --      ProgramBreak('In TList.SetEltType');
12 605 1-      END;
12 606 --
12 607 --      { Assign the field }
12 608 --      fEltClass := s;
12 609 --      fObjPtr := _GetA5 + FindObjID(s);
12 610 0- A  END;
12 611 --      {$ENDC}
12 612 --      {$%-}
12 613 --
12 614 --
12 615 --      {$IFC qDebug}
12 616 --      {$S MAFields}
12 617 --      {-----+
12 618 --      |   Fields                               |
12 619 --      +-----}
12 620 -- A  PROCEDURE TList.Fields (PROCEDURE DoToField (fieldName: Str255;
12 621 --      fieldAddr: Ptr;
12 622 --      fieldType: INTEGER)); OVERRIDE;
12 623 --
12 624 --      VAR
12 625 --      i:      INTEGER;
12 626 --      aString:  Str255;
12 627 0- A  BEGIN
12 628 --      DoToField('TList', NIL, bClass);
12 629 --      DoToField('fSize', @fSize, bInteger);
12 630 --      DoToField('fFirstOffset', @fFirstOffset, bLongInt);
12 631 --      DoToField('fDeletions', @fDeletions, bInteger);
12 632 --      DoToField('fEachLevel', @fEachLevel, bInteger);
12 633 --      DoToField('fEltClass', @fEltClass, bString);
12 634 --      DoToField('fObjPtr', @fObjPtr, bHexLongInt);
12 635 --      FOR i := 1 TO Min(fSize, 50) DO
12 636 1-      BEGIN
12 637 --      NumToString(i, aString);
12 638 --      aString := CONCAT('At[', aString, ']');
12 639 --      DoToField(aString, Ptr(ComputeAddress(i)), bObject);
12 640 1-      END;
12 641 --      INHERITED Fields(DoToField);
12 642 0- A  END;
12 643 --      {$ENDC}
12 644 --
12 645 --
12 646 --      {$IFC qInspector}
12 647 --      {$S MAInspector}
12 648 --      {-----+
12 649 --      |   GetInspectorName                     |
12 650 --      +-----}
12 651 -- A  PROCEDURE TList.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
12 652 --
12 653 0- A  BEGIN
12 654 --      IF fEltClass <> '' THEN
12 655 --      inspectorName := CONCAT('Of ', fEltClass);
12 656 0- A  END;
12 657 --      {$ENDC}
12 658 --
12 659 --
12 660 --      {$S MAnever}
12 661 --      {-----+
12 662 --      |   Compare                               |
12 663 --      +-----}
12 664 -- A  FUNCTION TSortedList.Compare (item1, item2: TObject): INTEGER;
12 665 --
12 666 0- A  BEGIN
12 667 --      {$IFC qDebug}
12 668 --      ProgramBreak('TSortedList.Compare must be overridden');
12 669 --      {$ENDC}
12 670 0- A  END;
12 671 --
12 672 --
12 673 --      {$S ListRes}
12 674 --      {-----+
12 675 --      |   GetItemNumber                       |
12 676 --      +-----}
12 677 -- A  FUNCTION TSortedList.GetItemNumber (item: TObject): INTEGER; OVERRIDE;
12 678 --
12 679 --      VAR
12 680 --      index:      INTEGER;
12 681 --      p:          LONGINT;
12 682 --      low,
12 683 --      high:      INTEGER;
12 684 --      compareResult:  INTEGER;
12 685 --
12 686 0- A  BEGIN
12 687 --      IF fSize = 0 THEN
12 688 --      index := 1
12 689 --      ELSE IF fDeletions > 0 THEN
12 690 --      index := INHERITED GetItemNumber(item)
12 691 --      ELSE
12 692 1-      BEGIN

```

```

12 693 --         low := 1;
12 694 --         high := fsize;
12 695 2-         REPEAT
12 696 --             index := BSR(low + high, 1);          { (low + high) DIV 2 }
12 697 --             compareResult := Compare(item, At(index));
12 698 --             IF compareResult < 0 THEN
12 699 --                 high := index - 1
12 700 --             ELSE
12 701 --                 low := index + 1;
12 702 -2         UNTIL (compareResult = 0) | (low > high);
12 703 --         IF compareResult > 0 THEN
12 704 --             index := index + 1;
12 705 -1         END;
12 706 --
12 707 --         GetItemNumber := index;
12 708 -0 A     END;
12 709 --
12 710 --
12 711 --         {$S ListRes}
12 712 --         {-----+
12 713 --         |   Insert   |
12 714 --         +-----}
12 715 -- A     PROCEDURE TSortedList.Insert (item: TObject);
12 716 --
12 717 0- A     BEGIN
12 718 --         InsertBefore(GetItemNumber(item), item);
12 719 -0 A     END;
12 720 --
12 721 --
12 722 --         {$S ListRes}
12 723 --         {-----+
12 724 --         |   Search   |
12 725 --         +-----}
12 726 -- A     FUNCTION TSortedList.Search (FUNCTION TestItem(anItem: TObject): INTEGER): TObject;
12 727 --
12 728 --         VAR
12 729 --             index:      INTEGER;
12 730 --             p:          LONGINT;
12 731 --             low,
12 732 --             high:       INTEGER;
12 733 --             theItem:    TObject;
12 734 --             compareResult: INTEGER;
12 735 --
12 736 -- B         FUNCTION IsThisTheItem (anItem: TObject): BOOLEAN;
12 737 0- B         BEGIN
12 738 --             IsThisTheItem := TestItem(anItem) = 0;
12 739 -0 B         END;
12 740 --
12 741 0- A     BEGIN
12 742 --         IF fsize > 0 THEN
12 743 --             IF fDeletions > 0 THEN
12 744 1-         BEGIN
12 745 --             Search := FirstThat(IsThisTheItem);
12 746 --             EXIT(Search);
12 747 -1         END
12 748 --         ELSE
12 749 1-         BEGIN
12 750 --             low := 1;
12 751 --             high := fsize;
12 752 --             WHILE low <= high DO
12 753 2-         BEGIN
12 754 --             index := BSR(low + high, 1);          { (low + high) DIV 2 }
12 755 --             theItem := At(index);
12 756 --             compareResult := TestItem(theItem);
12 757 --             IF compareResult = 0 THEN
12 758 3-         BEGIN
12 759 --                 Search := theItem;
12 760 --                 EXIT(Search);
12 761 -3         END
12 762 --             ELSE IF compareResult < 0 THEN
12 763 --                 high := index - 1
12 764 --             ELSE
12 765 --                 low := index + 1;
12 766 -2         END;
12 767 -1         END;
12 768 --
12 769 --         Search := NIL;
12 770 -0 A     END;
11 313 --
11 314 --     END.
11 315 --

```

```

13 1 --  { $P }
13 2 --  { UMacApp.p }
13 3 --  { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
13 4 --
13 5 --  { ??? Comments that contain '???' (such as this one) indicate issues that need
13 6 --    to be resolved. ??? }
13 7 --
13 8 --      { You should always override TApplication.DoMakeDocument.
13 9 --        You should always override TDocument.DoMakeViews.
13 10 --       You should always override TDocument.DoMakeWindows.
13 11 --       -
13 12 --     }
13 13 --
13 14 --
13 15 --  UNIT UMacApp;
13 16 --  { Unit UMacApp defines object types:
13 17 --    TEvtHandler      An abstract type that can handle events and commands.
13 18 --    TApplication      Its sole instance polls the user and network for events,
13 19 --                      and manages the desktop.
13 20 --    TDocument         Corresponds to a Finder document.  Manages document
13 21 --                      data in files and in main memory.
13 22 --    TWindow           Corresponds to a desktop Window.  The outermost of a
13 23 --                      nested set of frames.
13 24 --    TView             Renders an image of a document or other data, and
13 25 --                      identifies mouse-clicked objects.
13 26 --    TScroller         A view which can "scroll" its contents by effecting a
13 27 --                      coordinate transformation.
13 28 --    TPrintHandler     Handles printing at a specified resolution on behalf of
13 29 --                      a view.
13 30 --    TCommand          Handles a non-trivial user command, including undoing
13 31 --                      and mouse-tracking if required.
13 32 --    TDeskScrapView    The default view in the Clipboard, able to display only
13 33 --                      standard text and pictures.
13 34 --
13 35 --    as well as numerous other types, constants and global procedures for use by
13 36 --    MacApp and applications)
13 37 --
13 38 --  INTERFACE
13 39 --
13 40 --  {-----+
13 41 --  | Uses |
13 42 --  +-----+
13 43 --  USES
13 44 --      { $LOAD MacIntf.LOAD }
13 45 --      MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
13 46 --      { $LOAD }
13 47 --      Script,
13 48 --      UMAUtil,
13 49 --      UViewCoords,
13 50 --      UPatch,
13 51 --      UBusyCursor,
13 52 --      UFailure,
13 53 --      UMemory,
13 54 --      UMenuSetup,
13 55 --      UObject,
13 56 --      UList,
13 57 --      UAssociation
13 58 --  { $IFC qTrace }
13 59 --      , UTrace
13 60 --  { $ENDC }
13 61 --  { $IFC qWWNeeded }
13 62 --      , UWriteInWindow
13 63 --  { $ENDC }
13 64 --      ;
13 65 --
13 66 --  {-----+
13 67 --  | Constants |
13 68 --  +-----+
13 69 --  CONST
13 70 --      kCopyright      = 'Copyright © 1984, 1985, 1986, Apple Computer Inc.';
13 71 --
13 72 --  { M E N U S }
13 73 --      kMBarDisplayed   = 128; { Default 'MBAR' id of the menus that are
13 74 --                              read in and installed in menu bar }
13 75 --      kMBarNotDisplayed = 129; { Default 'MBAR' id of the menus that are
13 76 --                              read in but not installed }
13 77 --      kMBarHierarchical = 130; { Default 'MBAR' id of the menus that pop
13 78 --                              up when a menu item is chosen }
13 79 --
13 80 --  { C O M M A N D S }
13 81 --
13 82 --      { Predefined command numbers ??? Renumber these more logically ??? }
13 83 --      { Not all of these are caught by MacApp.
13 84 --        "<|" means MacApp catches it.
13 85 --        "<@" means TView catches it.
13 86 --        "<->" means the application must catch it if it is used as a
13 87 --          command number in any menu."
13 88 --
13 89 --      { Special command codes }
13 90 --
13 91 --      cNoCommand      =      0;      { <|> Command number representing no
13 92 --                                         command }
13 93 --
13 94 --      cCantUndo        =     -1;      { <|> Passed to TApplication.SetUndoText
13 95 --                                         to indicate that the command cannot
13 96 --                                         be undone. }
13 97 --
13 98 --  { Apple-menu commands }

```

```

13 99 --
13 100 --      cAboutApp      -      1;      { <|> "About <appname>_" }
13 101 --
13 102 --      { File-menu filing commands }
13 103 --
13 104 --      cNew            -      10;      { <|> "New" (??? reserve a range of NEW
13 105 --                                commands ???) }
13 106 --      cNewLast       -      19;
13 107 --      cOpen          -      20;      { <|> "Open _" (??? reserve a range of
13 108 --                                OPEN commands ???) }
13 109 --      cOpenLast      -      29;
13 110 --
13 111 --      cSave           -      30;      { <|> "Save" }
13 112 --      cClose         -      31;      { <|> "Close" }
13 113 --      cSaveAs        -      32;      { <|> "Save as_" }
13 114 --      cSaveCopy      -      33;      { <|> "Save a Copy in _" }
13 115 --      cRevert         -      34;      { <|> "Revert" (to previous version) }
13 116 --      cShowClipboard -      35;      { <|> "Show Clipboard"/"Hide Clipboard" }
13 117 --      cQuit           -      36;      { <|> "Quit" }
13 118 --
13 119 --
13 120 --      { Edit-menu commands }
13 121 --
13 122 --      { For the following command numbers, we must guarantee that
13 123 --      <command number> - cEditBase = <appropriate number to pass to
13 124 --      SystemEdit>. This relationship is enforced in
13 125 --      TApplication.IApplication. }
13 126 --      cEditBase       -      101;      { start of standard editing commands }
13 127 --      cUndo           -      101;      { <|> "Undo <command>"/"Redo <command>" }
13 128 --      cEditSep        -      102;      { line separating UNDO from CUT }
13 129 --      cCut           -      103;      { <|> "Cut" }
13 130 --      cCopy           -      104;      { <|> "Copy" }
13 131 --      cPaste          -      105;      { <|> "Paste" }
13 132 --      cClear          -      106;      { <-> "Clear" }
13 133 --      cEditLast       -      cClear;
13 134 --
13 135 --      cSelectAll      -      110;      { <|> "Select All" }
13 136 --
13 137 --      cTyping          -      120;      { for use in a TTypingCommand }
13 138 --      cMouseCommand   -      121;      { generic mouse command }
13 139 --
13 140 --      cStyleChange    -      130;      { "Menu" command for style change }
13 141 --      cRememberStyle  -      131;      { "Menu" command for remembering present style }
13 142 --
13 143 --      { Finder pseudo-commands }
13 144 --
13 145 --      cFinderNew      -      40;      { <|> The user selected the tool icon in
13 146 --                                the Finder and chose "Open" }
13 147 --      cFinderPrint    -      41;      { <|> The user selected document icons
13 148 --                                in the Finder and chose "Print" }
13 149 --      cFinderOpen     -      42;      { <|> The user selected document icons
13 150 --                                in the Finder and chose "Open" }
13 151 --
13 152 --
13 153 --      { File-menu printing commands }
13 154 --
13 155 --      cPrFileBase     -      176;      { Command numbers between cPrFileBase
13 156 --                                and cPrFileMax are sent to a
13 157 --                                document's fDocPrintHandler even if
13 158 --                                it is not in the fTarget chain }
13 159 --      cPrFileMax      -      195;
13 160 --
13 161 --      cPageSetup       -      176;      { <|> "Page Setup_" }
13 162 --      cPrintOne        -      177;      { <|> "Print One" }
13 163 --      cPrint           -      178;      { <|> "Print_" }
13 164 --      cPrintToFile    -      179;      { <|> "Print to file_" }
13 165 --
13 166 --      cPrintSpoolFile  -      190;      { <-> "Print spooled file_" }
13 167 --
13 168 --      cPrViewBase      -      201;      { Command numbers between cPrViewBase and
13 169 --                                cPrViewMax are printing commands
13 170 --                                applied to a displayed view in the
13 171 --                                fTarget chain }
13 172 --      cPrViewMax       -      250;
13 173 --
13 174 --      cShowBorders     -      199;      { <|> Toggle. "Show view borders"--also
13 175 --                                usable outside Debug menu }
13 176 --
13 177 --
13 178 --      { Zooming commands }      { Not being used at present }
13 179 --
13 180 --      cReduce50        -      301;      { <-> "Reduce 50%" }
13 181 --      cReduceToFit     -      302;      { <-> "Reduce to Fit" }
13 182 --      cShowFullSize    -      303;      { <-> "Show Full Size" }
13 183 --
13 184 --      { Control tracking }
13 185 --      cTrackingControl -      400;
13 186 --
13 187 --      { Debug-menu commands. Flag toggled by "cFoo" is called "gFoo" for any
13 188 --      "Foo" }
13 189 --
13 190 --      {$IFC qDebug}
13 191 --      cIdentifySoftware -      900;      { <|> Action. "Show Software Version"--
13 192 --                                WriteLn version info }
13 193 --      cExperimenting    -      901;      { <|> Toggle. "Enable experimental
13 194 --                                features" }
13 195 --      cReportMenuChoices -      902;      { <|> Toggle. "Trace menu commands"--
13 196 --                                WriteLn menu choices }
13 197 --      cIntenseDebugging -      904;      { <|> Toggle. "Intense Debugging" }

```

```

13 198 --      cTraceSetupMenus      -   905;   { <|> Toggle. "Allow Trace of Menu
13 199 --                                     Setups" if gTrace is also on }
13 200 --      cTraceIdle              -   906;   { <|> Toggle. "Allow Trace during Idle"
13 201 --                                     if gTrace is also on }
13 202 --      cDebugPrinting         -   912;   { <|> Toggle. "Debug Printing" }
13 203 --      cReportEvt            -   914;   { <|> Toggle. "Report Events"--WriteLn
13 204 --                                     events received from queue }
13 205 --      cDoFirstClick          -   915;   { <|> Toggle. "'Do First Click' for this
13 206 --                                     window" }
13 207 --      cVarClipPicSize        -   916;   { <|> Toggle. "Scale Pictures in
13 208 --                                     Clipboard to Window" }
13 209 --      cRefreshFrontWindow     -   917;   { <|> Action. "Refresh Front Window" }
13 210 --      cNewInspectorWindow    -   918;   { <|> Action. "Open new inspector window" }
13 211 --      cModalToggle          -   919;   { <|> Toggle. "Make front window modal/modeless"}
13 212 --      {SENDNC}
13 213 --      {$IFC qWWNeeded}
13 214 --          cDebugWind          -   913;   { <|> Action. "Show Debug Window" }
13 215 --      {SENDNC}
13 216 --
13 217 --      { D E B U G   F L A G S }
13 218 --
13 219 --      {$IFC NOT qDebug} { These start with 'g' because they are global variables when
13 220 --                        qDebug is TRUE }
13 221 --          gExperimenting        -           FALSE;
13 222 --          gReportMenuChoices    -           FALSE;
13 223 --          gIntenseDebugging     -           FALSE;
13 224 --          gDebugPrinting        -           FALSE;
13 225 --          gReportEvt            -           FALSE;
13 226 --
13 227 --          gMemMgtReport          -           FALSE;
13 228 --      {SENDNC}
13 229 --
13 230 --
13 231 --
13 232 --      { A L E R T S }
13 233 --
13 234 --      errReasonID              -   128;   { resource ID of errs resource describing
13 235 --                                     error messages }
13 236 --      errRecoveryID           -   129;   { resource ID of errs resource describing
13 237 --                                     recovery messages }
13 238 --      errOperationsID         -   130;   { resource ID of errs resource containing
13 239 --                                     operation strings }
13 240 --      errAppTable             -   1000;   { added to the MacApp resource ID to get
13 241 --                                     the ID of an application error table }
13 242 --
13 243 --      msgCmdErr                -   0;     { when added to a command number, gives
13 244 --                                     a message value to display the
13 245 --                                     alert "Could not complete the ...
13 246 --                                     command ..." }
13 247 --      msgAlert                 -   $FFFF0000;
13 248 --                                     { when added to an alert number, gives a
13 249 --                                     message value to display that alert }
13 250 --      msgLookup                -   $FFFE0000;
13 251 --                                     { when added to an INTEGER, gives a
13 252 --                                     message value that will look up the
13 253 --                                     INTEGER in table errOperationsID
13 254 --                                     and display the alert "Could not ...
13 255 --                                     because ..." }
13 256 --      msgAltRecovery           -   $FFFD0000;
13 257 --                                     { same as msgLookup, except the INTEGER
13 258 --                                     is used instead of the error number
13 259 --                                     to lookup the recovery string }
13 260 --
13 261 --      msgStrList               -   200 * $10000; { Standard list of operation
13 262 --                                     strings used by MacApp }
13 263 --      msgCancelled             -   msgStrList; { used with error = noErr if user
13 264 --                                     cancels Save, Quit, ... }
13 265 --      msgInitFailed            -   msgStrList + 1;
13 266 --      msgSaveFailed            -   msgStrList + 2;
13 267 --      msgRevertFailed         -   msgStrList + 3;
13 268 --      msgPrintFailed           -   msgStrList + 4;
13 269 --      msgNewFailed             -   msgStrList + 5;
13 270 --      msgOpenFailed            -   msgStrList + 6;
13 271 --      msgSaveAsFailed          -   msgStrList + 7;
13 272 --      msgSaveCopyFailed        -   msgStrList + 8;
13 273 --      msgDrawFailed            -   msgStrList + 9;
13 274 --      msgImportClipFailed      -   msgStrList + 10;
13 275 --      msgExportClipFailed      -   msgStrList + 11;
13 276 --
13 277 --
13 278 --      { error codes used in MacApp; applications should use errors between
13 279 --        -25000 and -29999 }
13 280 --
13 281 --      errSpooling              -   -20000;   { print spooling failed OSerr }
13 282 --      errRevertFNF            -   -20001;   { Revert failed because file was
13 283 --                                     deleted }
13 284 --      errFileChanged          -   -20002;   { disk file modification date
13 285 --                                     changed }
13 286 --      errSaveAgain            -   -20003;   { Save As / Save a Copy in an opened
13 287 --                                     document }
13 288 --      errFTypeChanged         -   -20004;   { disk file type changed }
13 289 --      errNoPrintDrvrr         -   -20005;   { print driver file not found }
13 290 --      errNotMyType            -   -20006;   { can't open this file type }
13 291 --
13 292 --      { Alerts 0 - 249 are reserved by MacApp. }
13 293 --
13 294 --      phGenError               -   100;     { 'Could not ^2, because ^0. ^1' }
13 295 --      phCmdErr                 -   101;     { 'Could not complete the ^^2" command because ^0. ^1'. }
13 296 --      phUnknownErr             -   102;     { 'Could not complete your request

```



```

13 297 -- because ^0. ^1; only used if no
13 298 -- operation is supplied. }
13 299 -- phSaveChanges - 110: { 'Save Changes Before Closing?' }
13 300 -- phRevert - 111: { 'Revert to last version saved?' }
13 301 -- phFileChanged - 112: { 'File changed since last save.' }
13 302 -- phPurgeOld - 113: { 'OK to purge old version before saving new one?' }
13 303 -- phReopenDoc - 114: { 'Document is already open.' }
13 304 -- phSpaceIsLow - 115: { 'Space is low' }
13 305 -- phUnsupportedROMs - 116: { 'Can't start app because your ROMs are too old.' }
13 306 -- phOfferReadOnly - 117: { 'Can't open for file for write. Open read-only?' }
13 307 --
13 308 -- phTooManyChars - 150: { 'Too Many Characters', used to reject
13 309 -- paste or keystrokes in UDialog and
13 310 -- UTEView }
13 311 --
13 312 -- phStylesTooBig - 151: { 'Too Many Styles', used to reject the
13 313 -- styles portion of a paste in UTEView }
13 314 --
13 315 -- phAboutApp - 201: { Can hold the 'About <appName>...'
13 316 -- message }
13 317 --
13 318 -- {$IFC qDebug}
13 319 -- phUnimplemented - 202: { 'This feature not yet implemented' }
13 320 -- {$ENDC}
13 321 --
13 322 -- { M A C A P P B U Z Z - S T R I N G R E S O U R C E }
13 323 -- kIDBuzzString - 250: { std list of buzzwords stored as
13 324 -- 'STR#' resources. This string list is reserved for MacApp; you
13 325 -- should choose a different resource ID for your strings. }
13 326 --
13 327 -- { Indices for strings in the buzz-string resource }
13 328 -- bzSaveAs - 1: { 'Save This Document As:' }
13 329 -- bzSaveCopy - 2: { 'Save a Copy In:' }
13 330 -- bzShowClip - 3: { 'Show Clipboard' }
13 331 -- bzHideClip - 4: { 'Hide Clipboard' }
13 332 -- bzUndo - 5: { 'Undo x' }
13 333 -- bzRedo - 6: { 'Redo x' }
13 334 -- bzCantDraw - 7: { 'Unable to draw contents of window' }
13 335 -- bzUntitled - 8: { name of untitled document; if blank
13 336 -- then this is taken from the window title template; if this includes
13 337 -- the string '<<<>>' then MacApp will substitute a number for the
13 338 -- string (e.g., if the buzz string contains 'Untitled-<<<>>' then
13 339 -- untitled windows will be named 'Untitled-1', 'Untitled-2, ... }
13 340 -- bzClosing - 9: { 'closing' }
13 341 -- bzQuitting - 10: { 'quitting' }
13 342 -- bzCantUndo - 11: { 'Can't Undo' }
13 343 -- bzSaveAnyways - 12: { 'save' }
13 344 -- bzRevertAnyways - 13: { 'revert' }
13 345 --
13 346 -- {$IFC qDebug}
13 347 -- { D E B U G B U Z Z - S T R I N G R E S O U R C E }
13 348 -- kDebugBuzzStrings - 251: { Debug buzzwords STR# resource }
13 349 -- bzMakeModal - 1: { 'Make Front Window Modal' }
13 350 -- bzMakeModeless - 2: { 'Make Front Window Modeless' }
13 351 -- bzDoFirstClick - 3: { 'Do First Click For This Window' }
13 352 -- bzDontDoFirstClick - 4: { 'Don't Do First Click For This Window' }
13 353 -- {$ENDC}
13 354 --
13 355 -- { R E S O U R C E I D S }
13 356 -- kIDClipWindow - 200: { window displaying the Clipboard }
13 357 -- {$IFC qWWNeeded}
13 358 -- kDebugParamsID - 300: { debug window }
13 359 -- {$ENDC}
13 360 --
13 361 --
13 362 -- { H I G H L I G H T I N G }
13 363 -- hloff - 1: { the selection is to be unhighlighted }
13 364 -- hldim - 2: { the selection is to be dimly highlighted }
13 365 -- hlon - 4: { the selection is to be fully highlighted }
13 366 --
13 367 -- { these constants can be used to test for combinations of fromHLtoHL,
13 368 -- when you do not care which is from and which is to }
13 369 -- hloffDim - hloff + hldim;
13 370 -- hldimOff - hloffDim;
13 371 -- hldimOn - hldim + hlon;
13 372 -- hlonDim - hldimOn;
13 373 -- hloffOn - hloff + hlon;
13 374 -- hlonOff - hloffOn;
13 375 --
13 376 --
13 377 --
13 378 -- { C H O I C E S }
13 379 -- mOKHit - 1;
13 380 -- mCancelHit - 2;
13 381 -- mButtonHit - 3;
13 382 -- mCheckBoxHit - 4;
13 383 -- mClusterHit - 5;
13 384 -- mEditTextHit - 6;
13 385 -- mIconHit - 7;
13 386 -- mListItemHit - 8;
13 387 -- mListScrollBarHit - 9;
13 388 -- mPictureHit - 10;
13 389 -- mPopupHit - 11;
13 390 -- mRadioHit - 12;
13 391 -- mStaticTextHit - 13;
13 392 -- mHScrollBarHit - 14;
13 393 -- mVScrollBarHit - 15;
13 394 -- mEditTabKey - 16;
13 395 -- mEditReturnKey - 17;

```

```

13 396 -- mEditEnterKey - 18;
13 397 -- mPatternHit - 19;
13 398 -- mControlHit - 20;
13 399 --
13 400 -- kNoIdentifier - ' ';
13 401 -- kNoTemplate - -1;
13 402 -- kNoResource - -1;
13 403 --
13 404 -- { M I S C E L L A N E O U S }
13 405 --
13 406 -- kYesButton - 1; { the 'Yes' button in various dialogs }
13 407 -- kNoButton - 3; { the 'No' button in various dialogs }
13 408 --
13 409 -- kLowSpaceInterval - 2*60*60; { default low space msg interval (2 seconds) }
13 410 --
13 411 -- kStdScrollUnit - 16; { default value of fScrollUnit used by scrollers }
13 412 --
13 413 -- kSBarSize - 16;
13 414 -- kSBarSizeMinus1 - kSBarSize - 1;
13 415 -- { !!! Prefer the above names }
13 416 -- kStdSzSBar - 16; { width/height of a standard
13 417 -- vertical/horizontal scroll bar }
13 418 -- kStdSzMinus1SBar - kStdSzSBar - 1;
13 419 --
13 420 -- kStdStaggerAmount - 16; { standard # of pixels to stagger windows }
13 421 --
13 422 -- kPrintInfoSize - 120; { size in bytes of printInfo record }
13 423 --
13 424 -- { size in bytes of the overhead for a resource file;
13 425 -- kRsrcFileOverhead is added in TDocument.DoNeedDiskSpace if the
13 426 -- document uses the rsrc fork.
13 427 -- You should add kRsrcTypeOverhead for each unique type you use
13 428 -- plus kRsrcOverhead for each resource you use
13 429 -- plus the total bytes in all the resources
13 430 -- plus the length of all the names for named resources. }
13 431 -- kRsrcFileOverhead - 256 + { resource header & stuff }
13 432 -- 30 { fixed part of resource map };
13 433 -- kRsrcTypeOverhead - 8; { overhead for each resource type }
13 434 -- kRsrcOverhead - 16; { overhead for each resource }
13 435 --
13 436 -- { ??? } kViewRsrcExpandAmt - $400; { ??? Is 1K enough for expansion, or too much maybe??? }
13 437 --
13 438 --
13 439 -- { $IFC qWWNeeded }
13 440 -- { debugging info }
13 441 -- kDebugFont - monaco; { Font for Debug Window }
13 442 -- kDebugSize - 9; { Font size for Debug Window }
13 443 -- { $ENDC }
13 444 --
13 445 -- { argument of TDocument.DoMakeViews }
13 446 -- kForDisplay - FALSE;
13 447 -- kForPrinting - NOT kForDisplay;
13 448 --
13 449 -- { arguments of TFrame.IScroller }
13 450 -- kWantHScrollBar - TRUE;
13 451 -- kWantVScrollBar - TRUE;
13 452 --
13 453 -- { arguments of TDocument.Save }
13 454 -- kAskForFilename - TRUE;
13 455 -- kMakingCopy - TRUE;
13 456 -- kSwitchToTarget - NOT kMakingCopy;
13 457 --
13 458 -- { argument of NewPaletteWindow }
13 459 -- kLeftPalette - h;
13 460 -- kTopPalette - v;
13 461 --
13 462 -- { argument of TApplication.SetUndoText }
13 463 -- kShowUndo - TRUE;
13 464 -- kShowRedo - FALSE;
13 465 -- kShowCantUndo - kShowUndo;
13 466 --
13 467 -- { argument of TDocument.IDocument }
13 468 -- kUsesDataFork - TRUE;
13 469 -- kUsesRsrcFork - TRUE;
13 470 -- kDataOpen - TRUE;
13 471 -- kRsrcOpen - TRUE;
13 472 --
13 473 -- { argument of TDocument.OpenDocFile }
13 474 -- kKeepingOpen - TRUE;
13 475 -- kReadingFile - TRUE;
13 476 --
13 477 -- kVisible - TRUE;
13 478 -- kInvisible - FALSE;
13 479 --
13 480 -- kRedraw - TRUE; { redraw after a requested state change }
13 481 -- kDontRedraw - FALSE; { don't redraw after a requested state change }
13 482 --
13 483 -- { MultiFinder event message masks }
13 484 -- kSuspendOrResume - $01;
13 485 -- kMouseMovedMessage - $FA;
13 486 --
13 487 -- kMaxSignatures - 32;
13 488 --
13 489 -- {-----+
13 490 -- | Types |
13 491 -- +-----}
13 492 -- TYPE
13 493 -- Integer8 - 0..7; { For packed fields in view templates }
13 494 -- Integer256 - 0..255;

```

```

13 495 --
13 496 -- { E V E N T S }
13 497 --
13 498 -- PEventRecord = ^EventRecord; { EventRecord is a ToolBox type }
13 499 --
13 500 -- EventInfo = RECORD { stores more information about the event in an unpacked form }
13 501 -- thePEvent: PEventRecord; { pointer to the original
13 502 -- packed ToolBox event record }
13 503 -- theBtnState: BOOLEAN; { bits found in the
13 504 -- modifiers field of an EventRecord }
13 505 -- theCmdKey: BOOLEAN; { Was Command key depressed? }
13 506 -- theShiftKey: BOOLEAN; { Was Shift key depressed? }
13 507 -- theAlphaLock: BOOLEAN; { Was Alpha Lock depressed? }
13 508 -- theOptionKey: BOOLEAN; { Was Option key depressed? }
13 509 -- theControlKey: BOOLEAN; { Was Control key depressed? }
13 510 -- theAutoKey: BOOLEAN; { TRUE if this was an auto key event }
13 511 --
13 512 -- theClickCount: INTEGER; { 0 = event was not a mouse
13 513 -- down;
13 514 -- 1-N = # of multiple clicks }
13 515 --
13 516 -- END;
13 517 --
13 518 -- IdlePhase = (idleBegin, idleContinue, idleEnd);
13 519 -- TrackPhase = (trackPress, trackMove, trackRelease);
13 520 --
13 521 -- { V I E W C O O R D I N A T E S }
13 522 --
13 523 -- SizeDeterminer = { tells how a view's size is to be determined; specified
13 524 -- separately in each dimension. }
13 525 -- (sizeSuperview, { View is the same size as its superview }
13 526 -- sizeRelSuperview, { View size is relative to the superview's size }
13 527 -- sizePage, { View to be the size of one page }
13 528 -- sizeFillPages, { View to grow upward to fill an exact number of pages }
13 529 -- sizeVariable, { View size fluctuates according to app-specific criteria }
13 530 -- sizeFixed); { No special default handling of size issues }
13 531 --
13 532 --
13 533 -- { H I G H L I G H T I N G }
13 534 -- HLState = hlOff..hlOn;
13 535 --
13 536 --
13 537 -- { D O C U M E N T S }
13 538 -- ( How to save a document in place, if MacApp decides this is needed.
13 539 -- sipNever: never save on top of old file (ie., save will fail if
13 540 -- there is not enough disk space)
13 541 -- sipAlways: save on top of old file without asking user
13 542 -- sipAskUser: save on top, but only if user confirms
13 543 -- )
13 544 -- SIPChoice = (sipNever, sipAlways, sipAskUser);
13 545 --
13 546 --
13 547 -- { O T H E R }
13 548 -- PAppFile = ^AppFile; { AppFile is defined in the Std File (Package
13 549 -- Manager) section of Inside Macintosh }
13 550 --
13 551 -- HTypeList = ^PTypeList;
13 552 -- PTypeList = ^ArrTypeList;
13 553 -- ArrTypeList = ARRAY[1..8000] OF OSType;
13 554 --
13 555 --
13 556 --
13 557 -- { C L A S S E S }
13 558 --
13 559 -- ( The TEvtHandler type represents abstract objects that handle certain kinds
13 560 -- of events:
13 561 -- Key events: both key down and auto key events
13 562 -- menu events: both enabling menus and menu items, and processing
13 563 -- menu commands
13 564 -- Read to/write from Disk
13 565 -- Idle events: when there are no other events to handle
13 566 -- Actual events: EvtHandlers, through their method 'DoHandleEvent',
13 567 -- may intercept and handle actual ToolBox events
13 568 -- Termination: when the application quits
13 569 --
13 570 -- TEvtHandler are linked into a list with the most specific object (usually
13 571 -- a selection) at the head of the list. The global variable 'gTarget'
13 572 -- contains the head of the list. (When a window is deactivated, this
13 573 -- global variable is cached in the window object, and retrieved when it
13 574 -- is later activated.)
13 575 --
13 576 -- For these kinds of events, the target (gTarget) gets the first crack at
13 577 -- handling the event. The default implementation of the methods of
13 578 -- TEvtHandler is to pass the event to the next element of the list. )
13 579 --
13 580 -- +-----+
13 581 -- | TEvtHandler |
13 582 -- +-----+
13 583 -- TEvtHandler = OBJECT(TObject)
13 584 --
13 585 -- fNextHandler: TEvtHandler; { the next element of the list, or NIL }
13 586 -- fIdleFreq: LONGINT; { the minimum number of ticks that can
13 587 -- pass before my DoIdle is called.
13 588 -- 0=never call my DoIdle. }
13 589 -- fLastIdle: LONGINT; { the tick count the last time my DoIdle
13 590 -- was called. }
13 591 --
13 592 -- { Init & Free }
13 593 -- PROCEDURE TEvtHandler.IEvtHandler(itsNextHandler: TEvtHandler);

```

```

13 594 --      PROCEDURE TEvtHandler.Free; OVERRIDE;
13 595 --
13 596 --      { Termination }
13 597 --      PROCEDURE TEvtHandler.Terminate;
13 598 --
13 599 --      ($IFC qDebug)
13 600 --      { Debugging }
13 601 --      PROCEDURE TEvtHandler.IdentifySoftware;
13 602 --      FUNCTION TEvtHandler.Lookupsymbol(VAR sym: String$): LONGINT;
13 603 --      { Called by the debugger to translate a symbol into a number;
13 604 --      default passes to the fNextHandler. sym will be uppercased.
13 605 --      Return of -1 means symbol was not found. If sym = '?' then
13 606 --      writein a list of available symbols. }
13 607 --      PROCEDURE TEvtHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
13 608 --      fieldAddr: Ptr;
13 609 --      fieldType: INTEGER)); OVERRIDE;
13 610 --
13 611 --      ($ENDC)
13 612 --
13 613 --      { Events }
13 614 --      PROCEDURE TEvtHandler.DoIdle(phase: IdlePhase);
13 615 --
13 616 --      FUNCTION TEvtHandler.DoHandleEvent(nextEvent: PEventRecord;
13 617 --      VAR commandToPerform: TCommand): BOOLEAN;
13 618 --      { I Handle the event and return TRUE if I want it, else I don't
13 619 --      handle it, and return FALSE }
13 620 --
13 621 --      { Double/Triple Clicks }
13 622 --      FUNCTION TEvtHandler.DoMultiClick(lastDownPt, newDownPt: Point): BOOLEAN;
13 623 --      { Called by TApplication.CountClicks. Should return TRUE if the 2
13 624 --      points are close enough to be considered part of a double/triple
13 625 --      click. (Both points are in global coordinates.) This is only
13 626 --      called if the mouse down was within the proper time range of the
13 627 --      previous mouse up. Default is to pass message to the
13 628 --      nextEventHandler (if one exists), otherwise to require that the
13 629 --      sum of the x & y distances is <= 5. ??? choose a default ??? }
13 630 --
13 631 --
13 632 --      { Key Events }
13 633 --      FUNCTION TEvtHandler.DoKeyCommand(ch: Char; aKeyCode: INTEGER;
13 634 --      VAR info: EventInfo): TCommand;
13 635 --      { Handles all keystrokes except those with the command key held down. }
13 636 --
13 637 --      FUNCTION TEvtHandler.DoCommandKey (ch: CHAR; VAR info: EventInfo): TCommand;
13 638 --      { Handles command-key combinations only. }
13 639 --
13 640 --
13 641 --      { MenuEvents }
13 642 --      FUNCTION TEvtHandler.DoMenuCommand(aCmdNumber: CmdNumber): TCommand;
13 643 --      { Returns NIL if command does not change the document (also perform
13 644 --      the command), otherwise return a TCommand object. }
13 645 --
13 646 --      PROCEDURE TEvtHandler.DoSetupMenus;
13 647 --
13 648 --      ($IFC qTemplateViews)
13 649 --      { View Creation from Templates }
13 650 --      FUNCTION TEvtHandler.DoCreateViews (itsDocument: TDocument; parentView: TView; itsRsrcID: INTEGER): TView;
13 651 --
13 652 --      FUNCTION TEvtHandler.CreateAView (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr): TView;
13 653 --      ($ENDC)
13 654 --
13 655 --      { Miscellaneous }
13 656 --      FUNCTION TEvtHandler.HandlesPrintingCommands: BOOLEAN;
13 657 --
13 658 --      PROCEDURE TEvtHandler.InstallSelection(wasActive, beActive: BOOLEAN);
13 659 --      { The Target is told to do this after a window is activated or
13 660 --      deactivated }
13 661 --
13 662 --      END;
13 663 --
13 664 --
13 665 --      (* The purpose of TApplication is to implement the main event loop that
13 666 --      all Macintosh applications must have. There is only one instance of
13 667 --      TApplication. Your main program should look similar to the following:
13 668 --
13 669 --      InitToolbox(, ,); { essential Toolbox initialization &
13 670 --      other one-time initialization }
13 671 --      NewObject(myApplication);
13 672 --      myApplication.Run;
13 673 --
13 674 --      One thing that TApplication does is to interpret the raw events that are
13 675 --      posted and convert them into higher-level events. For example, it
13 676 --      converts a click in the menu bar into a 'menu event' that contains the
13 677 --      menu ID/item number that was chosen by the user. *)
13 678 --
13 679 --      {-----+
13 680 --      | TApplication |
13 681 --      +-----+
13 682 --      TApplication = OBJECT(TEvtHandler)
13 683 --
13 684 --      { There are no TApplication variabes---they are global variables }
13 685 --
13 686 --      { Init & Free }
13 687 --      PROCEDURE TApplication.IApplication(itsMainFileType: OSType);
13 688 --      { itsMainFileType should be the the 4-byte file type that this
13 689 --      application reads/writes. We require this here so that
13 690 --      programmers do not have to explicitly override
13 691 --      TApplication.SFGetParms and supply the type there. That is the
13 692 --      only place where the main file type is used. }

```

```

13 693 --
13 694 --      ($IFC qDebug)
13 695 --      PROCEDURE TApplication.GetInspectorName (VAR inspectorName: Str255): OVERRIDE;
13 696 --
13 697 --      PROCEDURE TApplication.Fields (PROCEDURE DoToField (fieldName: Str255;
13 698 --                                fieldAddr: Ptr;
13 699 --                                fieldType: INTEGER)): OVERRIDE;
13 700 --
13 701 --      ($ENDC)
13 702 --
13 703 --      { Event Handling }
13 704 --      FUNCTION TApplication.GetEvent(eventMask: INTEGER; sleep: LONGINT; cursorRgn: RgnHandle;
13 705 --                                VAR anEvent: EventRecord): BOOLEAN;
13 706 --      { Calls WaitNextEvent (or GetNextEvent) with the given event mask, sleep, and cursor
13 707 --        region parameters. You can override this if you have an alternate source for events. }
13 708 --
13 709 --      FUNCTION TApplication.HandleAlienEvent(VAR theEventInfo: EventInfo): TCommand;
13 710 --      { This method is called upon to deal with network events and
13 711 --        'application-specific' events }
13 712 --
13 713 --      PROCEDURE TApplication.MainEventLoop;
13 714 --
13 715 --      PROCEDURE TApplication.DispatchEvent(VAR theEventInfo: EventInfo; VAR commandToPerform: TCommand);
13 716 --
13 717 --      PROCEDURE TApplication.HandleEvent (VAR theEvent: EventRecord);
13 718 --      { When this is called by MacApp, theEvent is a pointer to the global
13 719 --        variable gLastEvent. In special circumstances, you might
13 720 --        'fake' an event record and pass a pointer to it. }
13 721 --
13 722 --      PROCEDURE TApplication.PostHandleEvent (VAR theEventInfo: EventInfo);
13 723 --
13 724 --      FUNCTION TApplication.HandleActivateEvent (VAR theEventInfo: EventInfo): TCommand;
13 725 --      { Handles activate (and deactivate) events. Calls the
13 726 --        window's Activate method, and sets gRedrawMenuBar to true if
13 727 --        activating the window. Returns gNoChanges. }
13 728 --
13 729 --      FUNCTION TApplication.HandleDiskEvent (VAR theEventInfo: EventInfo): TCommand;
13 730 --      { Handles disk events. Calls DIBadMount if the high word of
13 731 --        theEvent's message is <> noErr. Returns gNoChanges. }
13 732 --
13 733 --      FUNCTION TApplication.HandleKeyDownEvent (VAR theEventInfo: EventInfo): TCommand;
13 734 --      { Handles key down events. If the command key is down, then this
13 735 --        calls CommandKey. Otherwise it calls gTarget.DoKeyCommand.
13 736 --        Returns the result of CommandKey or DoKeyCommand. }
13 737 --
13 738 --      FUNCTION TApplication.HandleMouseDown(VAR theEventInfo: EventInfo): TCommand;
13 739 --      { Called when a mouse down occurs; this cases on whereMouseDown
13 740 --        and does the right thing. }
13 741 --
13 742 --      FUNCTION TApplication.TrackMouse (globalMouse, hysteresis: Point;
13 743 --                                theCommand: TCommand): TCommand;
13 744 --
13 745 --      FUNCTION TApplication.HandleMouseUp (VAR theEventInfo: EventInfo): TCommand;
13 746 --      { Sets gLastUptime and sets gEventInfo.theClickCount to gClickCount.
13 747 --        Returns gNoChanges. }
13 748 --
13 749 --      FUNCTION TApplication.HandleSystemEvent (VAR theEventInfo: EventInfo): TCommand;
13 750 --      { Handles "system" events, in particular suspend/resume events and mouse
13 751 --        moved events. Calls RegainControl if switching in, AboutToLoseControl
13 752 --        if switching out, or TrackCursor if it is a mouse-moved event.
13 753 --        Returns gNoChanges. }
13 754 --
13 755 --      FUNCTION TApplication.HandleUpdateEvent (VAR theEventInfo: EventInfo): TCommand;
13 756 --      { Handles window update events. Calls UpdateEvent for the window
13 757 --        indicate by the event, and returns gNoChanges. }
13 758 --
13 759 --      FUNCTION TApplication.IsDeskAccessory (aWmgrWindow: WindowPtr): BOOLEAN;
13 760 --      { Returns true if the given Window Manager window is a desk accessory.
13 761 --        We assume that if the window has a negative windowKind then it
13 762 --        is a desk accessory. }
13 763 --
13 764 --      PROCEDURE TApplication.OpenDeskAccessory (deskAccName: Str255);
13 765 --      { Attempts to open the desk accessory whose name is deskAccName.
13 766 --        Called from TApplication.MenuEvent when an Apple menu item
13 767 --        is chosen, and the item has a negative command number. }
13 768 --
13 769 --      PROCEDURE TApplication.PollEvent;
13 770 --
13 771 --      PROCEDURE TApplication.Run;
13 772 --      { HandleFinderRequest, and if not finder printing, call
13 773 --        MainEventLoop }
13 774 --
13 775 --      PROCEDURE TApplication.UpdateAllWindows;
13 776 --      { Process all update events in the queue. }
13 777 --
13 778 --      FUNCTION TApplication.InModalState: BOOLEAN;
13 779 --      { Returns true if the front window is modal. }
13 780 --
13 781 --      FUNCTION TApplication.InModalMenuState: BOOLEAN;
13 782 --      { Returns true if the front window does not allow menu commands. }
13 783 --
13 784 --      ($IFC qDebug)
13 785 --      PROCEDURE TApplication.ReportEvent(VAR theEventInfo: EventInfo);
13 786 --      { Displays information about an event in the debugging window. }
13 787 --
13 788 --      PROCEDURE TApplication.ReportMouseDown(where: INTEGER);
13 789 --      { Displays the location of the mouse in terms of a FindWindow result. }
13 790 --
13 791 --      ($ENDC)

```

```

13 792 --
13 793 --      ( Target and Cohandlers )
13 794 --      PROCEDURE TApplication.InstallCohandler(aCohandler: TEvtHandler;
13 795 --          addIt: BOOLEAN);
13 796 --          { Used to add (addIt = TRUE) or delete (addIt = FALSE) a cohandler
13 797 --            to/from the global list of cohandlers }
13 798 --
13 799 --      PROCEDURE TApplication.EachHandler(aFirstHandler: TEvtHandler;
13 800 --          PROCEDURE DoToEvtHandler(anEvtHandler: TEvtHandler));
13 801 --          { Performs 'DoToEvtHandler' to aFirstHandler, then to its fNextHandler,
13 802 --            etc., onward until the fNextHandler chain ends at NIL }
13 803 --
13 804 --      FUNCTION TApplication.FirstHandlerThat(aFirstHandler: TEvtHandler;
13 805 --          FUNCTION TestEvtHandler(anEvtHandler: TEvtHandler): BOOLEAN):
13 806 --          TEvtHandler;
13 807 --          { Calls TestEvtHandler for each event handler until TestEvtHandler
13 808 --            returns true. Returns the TEvtHandler object for which
13 809 --            TestEvtHandler returned true, or NIL if TestEvtHandler never
13 810 --            returned true. }
13 811 --
13 812 --      PROCEDURE TApplication.SetTarget (newTarget: TEvtHandler);
13 813 --          { Called to set the current target for the application. }
13 814 --
13 815 --      ( Finder Requests )
13 816 --      FUNCTION TApplication.CanOpenDocument(itsCmdNumber: CmdNumber;
13 817 --          VAR anAppFile: AppFile): BOOLEAN;
13 818 --          { Simulates the filtering done by Std File; this is only called when
13 819 --            opening/printing documents from the finder; when using
13 820 --            Std File, Std File does the filtering. }
13 821 --
13 822 --      PROCEDURE TApplication.HandleFinderRequest;
13 823 --          { Gets the info from the finder about what files to open/print and
13 824 --            opens/prints them. }
13 825 --
13 826 --      ( Opening / Printing Documents )
13 827 --      FUNCTION TApplication.AlreadyOpen(fileName: Str255;
13 828 --          volRefnum: INTEGER): TDocument;
13 829 --          { Given a name/volRefnum, if that document is already opened,
13 830 --            returns the TDocument. Otherwise returns NIL. }
13 831 --
13 832 --      FUNCTION TApplication.ChooseDocument(itsCmdNumber: CmdNumber;
13 833 --          VAR anAppFile: AppFile): BOOLEAN;
13 834 --          { Call this to make a Std File Get call; returns TRUE is
13 835 --            user selected a file. }
13 836 --
13 837 --      FUNCTION TApplication.DoMakeDocument(itsCmdNumber: CmdNumber): TDocument;
13 838 --          { Must be overridden. Based on itsCmdNumber create a document
13 839 --            object of the appropriate kind. }
13 840 --
13 841 --      FUNCTION TApplication.KindOfDocument(itsCmdNumber: CmdNumber;
13 842 --          itsPAppFile: PAppFile): CmdNumber;
13 843 --          { Given a cmd number and a specification of the file, return the cmd
13 844 --            number to pass to DoMakeDocument that indicates the type of
13 845 --            document to make; Default is to return itsCmdNumber. If you
13 846 --            have multiple document types, a good convention is to return
13 847 --            the cmd numbers assigned to create new documents of each kind.
13 848 --            itsPAppFile will be NIL if we are creating a brand new document,
13 849 --            rather than opening an existing document. }
13 850 --
13 851 --      PROCEDURE TApplication.OpenNew(itsCmdNumber: CmdNumber);
13 852 --          { Called when the application is opened (itsCmdNumber = cFinderNew),
13 853 --            or when NEW is chosen from menu (itsCmdNumber = cNew). If you
13 854 --            do not want to create a new document when the user opens the
13 855 --            application, override this and do nothing if itsCmdNumber =
13 856 --            cFinderNew. }
13 857 --
13 858 --      PROCEDURE TApplication.OpenOld(itsOpenCmd: CmdNumber;
13 859 --          anAppFile: AppFile);
13 860 --          { Called when opening an existing document from finder (itsCmdNumber
13 861 --            = cFinderOpen) or if OPEN is chosen from menu (itsCmdNumber =
13 862 --            cOpen). }
13 863 --
13 864 --      FUNCTION TApplication.PrintDocument(anAppFile: AppFile): BOOLEAN;
13 865 --          { Called to print a document from the finder. Returns TRUE if the
13 866 --            user did not cancel printing of the rest of the documents. }
13 867 --
13 868 --      PROCEDURE TApplication.SFGetParms(itsCmdNumber: CmdNumber;
13 869 --          VAR dlgID: INTEGER; VAR where: Point;
13 870 --          VAR fileFilter, dlgHook, filterProc: ProcPtr;
13 871 --          typeList: HTypeList);
13 872 --          { Called to return all the parameters that should be passed to
13 873 --            SFGetFile; when called, typeList is a valid handle to a
13 874 --            0-length block.
13 875 --            Defaults to
13 876 --            dlgID = getDlgID;
13 877 --            where = (100, 100);
13 878 --            fileFilter = NIL;
13 879 --            dlgHook = NIL;
13 880 --            filterProc = NIL;
13 881 --            typeList = list of just the main file type supported by the
13 882 --            application. (If type list ends up with nothing in it,
13 883 --            then all file types are supported. )
13 884 --
13 885 --      ( Termination )
13 886 --      PROCEDURE TApplication.Close;
13 887 --          { Called when the user chooses Quit from the menu, and tries to save
13 888 --            all the open documents. If all succeed the application
13 889 --            terminates. Signals Failure with error = noErr and message =
13 890 --

```

```

13 891 --      msgCancelled if user cancels )
13 892 --
13 893 --      { Hierarchy }
13 894 --      PROCEDURE TApplication.AddDocument(aNewDocument: TDocument);
13 895 --      { Add another document to my list of documents, and make it the
13 896 --      current one }
13 897 --      PROCEDURE TApplication.AddFreeWindow(aWindow: TWindow);
13 898 --      { Add the window to the free-window list }
13 899 --      PROCEDURE TApplication.DeleteDocument(docToDelete: TDocument);
13 900 --      { Delete a document from my list of documents, and adjust gDocument
13 901 --      and activeFrame if needed }
13 902 --      PROCEDURE TApplication.DeleteFreeWindow(windowToDelete: TWindow);
13 903 --      { Delete a window from my list of free windows }
13 904 --      PROCEDURE TApplication.ForEachFreeWindow(PROCEDURE DoToWindow(aWindow: TWindow));
13 905 --      { Perform DoToWindow on each TWindow object in the list
13 906 --      gFreeWindowList }
13 907 --      PROCEDURE TApplication.ForAllDocumentsDo(PROCEDURE DoToDoc(aDocument: TDocument));
13 908 --      { Perform the given procedure on all open documents currently owned
13 909 --      by the application }
13 910 --      PROCEDURE TApplication.ForAllWindowsDo(PROCEDURE DoToWind(aWindow: TWindow));
13 911 --      { Perform the given procedure on all windows of all documents, as
13 912 --      well as on all free-standing (document-less) windows }
13 913 --
13 914 --
13 915 --      { Window Manager Windows }
13 916 --      PROCEDURE TApplication.SelectWMgrWindow (aWMgrWindow: WindowPtr);
13 917 --      { "Selects" aWMgrWindow by calling the toolbox routine SelectWindow. When any
13 918 --      window is selected, it goes through here. We never call SelectWindow
13 919 --      directly.}
13 920 --
13 921 --      FUNCTION TApplication.GetRsrcWindow(storage: Ptr; rsrcId: INTEGER;
13 922 --      VAR isResizable, isClosable: BOOLEAN): WindowPtr;
13 923 --      { Get a Window Manager window resource whose resource id is rsrcId.
13 924 --      Return isResizable and isClosable based upon the 'WIND'
13 925 --      resource. }
13 926 --
13 927 --      FUNCTION TApplication.WMgrToWindow (aWMgrWindow: WindowPtr): TWindow;
13 928 --      { Returns the window object that represents the given Window Manager
13 929 --      window, or NIL if there is no window object. }
13 930 --
13 931 --
13 932 --      { Idle Events }
13 933 --      PROCEDURE TApplication.Idle(phase: IdlePhase);
13 934 --
13 935 --
13 936 --      { Cursor }
13 937 --      FUNCTION TApplication.TrackCursor: BOOLEAN;
13 938 --      { Returns true if the cursor is set by a view. Otherwise the cursor is set
13 939 --      to an arrow and TrackCursor returns false.}
13 940 --
13 941 --
13 942 --      { Menu Events }
13 943 --      FUNCTION TApplication.DoCommandKey(ch: CHAR; VAR info: EventInfo): TCommand; OVERRIDE;
13 944 --      { Called when a keyDown event is received and the command key is down. }
13 945 --
13 946 --      FUNCTION TApplication.MenuEvent(menuItem: LONGINT): TCommand;
13 947 --      { Given a value returned by MenuKey or MenuSelect, figure out the
13 948 --      command number that was chosen, have the application create a
13 949 --      command object, and return it. }
13 950 --
13 951 --      PROCEDURE TApplication.SetUndoText(cmdDone: BOOLEAN;
13 952 --      aCmdNumber: CmdNumber);
13 953 --      { Called to setup the Undo menu item: if cmdDone is TRUE it reads
13 954 --      Undo, otherwise it reads Redo. aCmdNumber indicates the
13 955 --      command that goes after the Undo/Redo. }
13 956 --
13 957 --      PROCEDURE TApplication.SetupTheMenus;
13 958 --      { Initiates the process of enabling & checking menu items. }
13 959 --
13 960 --
13 961 --      { Opening and Closing Windows }
13 962 --      PROCEDURE TApplication.CloseWMgrWindow(aWMgrWindow: WindowPtr);
13 963 --      { Called when user closes a window either with a menu item or with
13 964 --      a GoAway box. Signals Failure with error = noErr and message =
13 965 --      msgCancelled if user cancels }
13 966 --
13 967 --
13 968 --      { Command Management }
13 969 --      PROCEDURE TApplication.CommitLastCommand;
13 970 --
13 971 --      PROCEDURE TApplication.PerformCommand(command: TCommand);
13 972 --      { This DOES NOT check to see if command is gNoChanges. }
13 973 --
13 974 --
13 975 --      { Double/Triple Clicks }
13 976 --      FUNCTION TApplication.CountClicks(aPDownEvent: PEventRecord;
13 977 --      whereMouseDown: INTEGER): INTEGER;
13 978 --      { Called by TApplication.HandleMouseDown. Returns the number of
13 979 --      clicks that can be considered multiple clicks (e.g. if it
13 980 --      returns 1 the mouse down should be treated as a single click, or
13 981 --      possibly the first click of a multi-click sequence. If it returns
13 982 --      2 the click should be considered a double-click, etc. A
13 983 --      click is considered part of a multi-click sequence if the
13 984 --      mouse down was within the proper time range of the previous
13 985 --      mouse up, and was within the proper number of pixels of the last
13 986 --      mouse down. TApplication.HandleMouseDown sets theClickCount
13 987 --      of its EventInfo record to the result of this function.
13 988 --
13 989 --      aPDownEvent is a pointer to the mouse down event.

```

```

13 990 --      whereMouseDown is the result of FindWindow on the event. }
13 991 --
13 992 --
13 993 --      { Command Handlers }
13 994 --      FUNCTION TApplication.DoMenuCommand(aCmdNumber: CmdNumber): TCommand;
13 995 --      OVERRIDE;
13 996 --
13 997 --      PROCEDURE TApplication.DoSetupMenus: OVERRIDE;
13 998 --
13 999 --
13 1000 --      { Clipboard stuff } { NB: Perhaps the categorization of TApp methods needs
13 1001 --      redoing }
13 1002 --      PROCEDURE TApplication.AbandonUndoClipboard;
13 1003 --      { Called when the undo clipboard is no longer needed. Calls
13 1004 --      gClipUndoView.FreeFromClipboard and sets gClipUndoView to NIL. }
13 1005 --
13 1006 --      PROCEDURE TApplication.AboutToLoseControl(convertClipboard: BOOLEAN);
13 1007 --      { Called when about to activate a Desk Accessory, switch out to
13 1008 --      another application, or Terminate. convertClipboard is generally
13 1009 --      true, except it will be FALSE if we get a Swticher event that does
13 1010 --      not require Clipboard conversion }
13 1011 --
13 1012 --      PROCEDURE TApplication.AbsorbScrapStuff;
13 1013 --      { Called to retrieve the current InfoScrap record from low memory.
13 1014 --      This will be used to determine if the scrap has changed. }
13 1015 --
13 1016 --      PROCEDURE TApplication.CheckDeskScrap;
13 1017 --      { Checks to see if the desk scrap has changed by calling
13 1018 --      AbsorbScrapStuff and comparing the current scrap change count
13 1019 --      with the previous change count. If the scrap has changed, then
13 1020 --      the current clipboard becomes the undo clipboard and
13 1021 --      ReadFromDeskScrap is called to read in the new desk scrap. }
13 1022 --
13 1023 --      PROCEDURE TApplication.ClaimClipboard(clipView: TView);
13 1024 --      { Makes clipView the new view displayed by the Clipboard. Typically
13 1025 --      called by a Cut or Copy command. }
13 1026 --
13 1027 --      FUNCTION TApplication.GetDataToPaste(aDataHandle: Handle;
13 1028 --      VAR dataType: ResType): LONGINT;
13 1029 --      { Returns the number of bytes in the paste data. Will signal
13 1030 --      Failure if it gets an error. }
13 1031 --
13 1032 --      PROCEDURE TApplication.LaunchClipboard;
13 1033 --      { Called by TApplication.Run, before the main event loop. }
13 1034 --
13 1035 --      FUNCTION TApplication.MakeClipboardWindow: TWindow;
13 1036 --      { Called by TApplication.TApplication to make the clipboard window
13 1037 --      and associated frame(s). clipView is the view to be initially
13 1038 --      installed in the clipboard--it is gClipOrphanage. }
13 1039 --
13 1040 --      PROCEDURE TApplication.ReadFromDeskScrap;
13 1041 --      { Called at launch and when the desk scrap changes. Calls
13 1042 --      TApplication.MakeViewForAlienClipboard, then ClaimClipboard }
13 1043 --
13 1044 --      FUNCTION TApplication.MakeViewForAlienClipboard: TView;
13 1045 --      { Returns gClipOrphanage, which is capable of displaying scrap
13 1046 --      of type TEXT or PICT. Override this to create a view to
13 1047 --      handle other types of scrap data. }
13 1048 --
13 1049 --      PROCEDURE TApplication.RegainControl(checkClipboard: BOOLEAN);
13 1050 --      { Called when switching in or when leaving a desk accessory }
13 1051 --
13 1052 --      PROCEDURE TApplication.SetClipView(clipView: TView);
13 1053 --      { Installs the given view into the clipboard window. }
13 1054 --
13 1055 --      PROCEDURE TApplication.SwapClipViews;
13 1056 --      { Swaps the undo clipboard view with the current clipboard view.
13 1057 --      Used during an Undo or Redo. }
13 1058 --
13 1059 --      ($IFC qDebug)
13 1060 --      { Debugging }
13 1061 --      PROCEDURE TApplication.IdentifySoftware: OVERRIDE;
13 1062 --      ($ENDC)
13 1063 --
13 1064 --      { Error Alerts }
13 1065 --      PROCEDURE TApplication.ShowError(error: OSerr; message: LONGINT);
13 1066 --      PROCEDURE TApplication.SpaceIsLow;
13 1067 --      { Called when space is low, at intervals of kLowSpaceInterval
13 1068 --      ticks. Displays an alerting informing the user that
13 1069 --      memory space is low. Override this if you wish to take
13 1070 --      more appropriate action. }
13 1071 --      END;
13 1072 --
13 1073 --
13 1074 --      +-----+
13 1075 --      | TDocument |
13 1076 --      +-----+
13 1077 --      TDocument = OBJECT(TEvtHandler)
13 1078 --
13 1079 --      fWindowList:      TList;      { list of windows belonging to this
13 1080 --                                   document }
13 1081 --      fViewList:        TList;      { list of views belonging to document }
13 1082 --      fDocPrintHandler: TPrintHandler; { the object to be told to PRINT or
13 1083 --                                   DoSetupMenus when 'Print',
13 1084 --                                   'Print One', or 'Page Setup...'
13 1085 --                                   is selected from the menu while
13 1086 --                                   this is the active document. }
13 1087 --
13 1088 --      fChangeCount:      LONGINT;    { master count of changes since last

```



```

13 1089 --          Save )
13 1090 --      fSavePrintInfo:    BOOLEAN;    ( if TRUE for a document saved on
13 1091 --                               disk, the 'print info' record
13 1092 --                               of the fDocPrintHandler will be
13 1093 --                               written out to the data fork
13 1094 --                               before other writing takes
13 1095 --                               place )
13 1096 --      fSharePrintInfo:    BOOLEAN;    ( if TRUE, then all printHandlers
13 1097 --                               associated with views belonging
13 1098 --                               to the same document will share
13 1099 --                               the same 'print info' record )
13 1100 --      fPrintInfo:         Handle;      ( if non-NIL, this is a handle to a
13 1101 --                               120-byte 'print info' record )
13 1102 --
13 1103 --      fTitle:              STRING[63];  ( file name )
13 1104 --      fFileType:           OSType;      ( file type )
13 1105 --      fCreator:            OSType;      ( creator ID )
13 1106 --      fVolRefNum:          INTEGER;     ( volume refNum )
13 1107 --      fModDate:            LONGINT;     ( file mod date when last read/saved )
13 1108 --      fReopenAlert:        BOOLEAN;     ( whether to give an alert if user
13 1109 --                               reopens doc )
13 1110 --      fSaveExists:         BOOLEAN;     ( whether a disk file representing
13 1111 --                               this document exists )
13 1112 --      fCommitOnSave:       BOOLEAN;     ( commit the last command before
13 1113 --                               saving the document, if it affected the
13 1114 --                               document. This defaults to TRUE, but
13 1115 --                               carefully written applications could set
13 1116 --                               this FALSE. )
13 1117 --      fUsesDataFork:       BOOLEAN;     ( whether the document uses the data
13 1118 --                               fork of the file )
13 1119 --      fUsesRsrcFork:       BOOLEAN;     ( whether the document uses the rsrc
13 1120 --                               fork of the file )
13 1121 --      fDataOpen:           BOOLEAN;     ( whether the data fork should be
13 1122 --                               open all the time )
13 1123 --      fRsrcOpen:           BOOLEAN;     ( whether the resources fork be open
13 1124 --                               all the time )
13 1125 --      fDataPerm:           INTEGER;     ( permission to use for reading data
13 1126 --                               fork )
13 1127 --      fRsrcPerm:           INTEGER;     ( permission to use for reading the
13 1128 --                               rsrc fork )
13 1129 --      fDataRefnum:         INTEGER;     ( refnum; valid only if data fork is
13 1130 --                               left open )
13 1131 --      fRsrcRefnum:         INTEGER;     ( refnum; valid only if resource
13 1132 --                               fork is left open )
13 1133 --                               ( these will be -1 if the fork is
13 1134 --                               not open or if the corresponding
13 1135 --                               fXXXOpen flag is FALSE )
13 1136 --      fSaveInPlace:        SIPChoice;   ( how to save files in place (if
13 1137 --                               desired) )
13 1138 --
13 1139 --      ( Init & Free )
13 1140 --      PROCEDURE TDocument.IDocument(itsFileType, itsCreator: OSType;
13 1141 --          usesDataFork, usesRsrcFork: BOOLEAN;
13 1142 --          keepsDataOpen, keepsRsrcOpen: BOOLEAN);
13 1143 --      PROCEDURE TDocument.Free; OVERRIDE;
13 1144 --          ( This does not call FreeData by default, since you may need to
13 1145 --            control the order in which things are freed. Your override
13 1146 --            of TDocument.Free can call FreeData if convenient. )
13 1147 --
13 1148 --      PROCEDURE TDocument.FreeData;
13 1149 --          ( Called when a document is reverted, in order to free it's data
13 1150 --            objects. You may also wish to call this from you document's
13 1151 --            Free method, if convenient. )
13 1152 --
13 1153 --      PROCEDURE TDocument.FreeFromClipboard;
13 1154 --          ( Called to free a Clipboard document. Simply calls Free. )
13 1155 --
13 1156 --      (SIFC qDebug)
13 1157 --      PROCEDURE TDocument.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
13 1158 --
13 1159 --      PROCEDURE TDocument.Fields (PROCEDURE DoToField (fieldName: Str255;
13 1160 --          fieldAddr: Ptr;
13 1161 --          fieldType: INTEGER)); OVERRIDE;
13 1162 --      ($ENDC)
13 1163 --
13 1164 --      ( Opening / Printing Documents )
13 1165 --      PROCEDURE TDocument.DoInitialState;
13 1166 --          ( Called for 'New', 'Revert' to blank, Open Tool. Default is to do
13 1167 --            nothing; You may need to do further initialization that you
13 1168 --            would not do in case of an Open... )
13 1169 --
13 1170 --      PROCEDURE TDocument.DoMakeWindows;
13 1171 --          ( Should take the views created by DoMakeViews and create windows
13 1172 --            and frames to display them. )
13 1173 --
13 1174 --      PROCEDURE TDocument.DoMakeViews(forPrinting: BOOLEAN);
13 1175 --          ( Create all necessary views for this document (based on
13 1176 --            forPrinting flag) and store into fields of your document. The
13 1177 --            document will then be sent either DoMakeWindows or Print. )
13 1178 --
13 1179 --      PROCEDURE TDocument.DoRead(aRefNum: INTEGER;
13 1180 --          rsrcExists, forPrinting: BOOLEAN);
13 1181 --      PROCEDURE TDocument.DoWrite(aRefNum: INTEGER; makingCopy: BOOLEAN);
13 1182 --          ( These just read/write the print state if necessary.
13 1183 --            rsrcExists indicates whether the document has a resource
13 1184 --            fork opened or not; it is FALSE if fUsesRsrcFork is FALSE.
13 1185 --            makingCopy indicates whether this save is for making a
13 1186 --            copy of the document (vs. a regular Save or Save As.);
13 1187 --            disk-based documents (especially) might optimize their

```

```

13 1188 --          behavior if this is TRUE. )
13 1189 --
13 1190 -- FUNCTION TDocument.HandlesPrintingCommands: BOOLEAN; OVERRIDE;
13 1191 --
13 1192 -- FUNCTION TDocument.OpenAFile(name: Str255; volRefnum: INTEGER;
13 1193 --   openData, openRsrc: BOOLEAN; dataPerm, rsrcPerm: INTEGER;
13 1194 --   VAR dataRefnum, rsrcRefnum: INTEGER): OSErr;
13 1195 --   { Called from ReadFromFile to open the document's data and/or
13 1196 --     resource forks.}
13 1197 --
13 1198 -- PROCEDURE TDocument.OpenAgain(itsCmdNumber: CmdNumber;
13 1199 --   openingDoc: TDocument);
13 1200 --   { Called if the user tries to open the same document twice.
13 1201 --     openingDoc will be the document object being opened. Default
13 1202 --     brings SELF to the front and calls Failure(noErr, 0), which
13 1203 --     will abort opening the document. (The goal is to prevent the
13 1204 --     user from having 2 copies of the same document open, in which
13 1205 --     case s/he could make incompatible changes to both.)
13 1206 --
13 1207 --     You might override this if you want to open a second window on the
13 1208 --     same document. In the ideal case, you should make both windows
13 1209 --     display exactly the same data, and changes made to one should
13 1210 --     be reflected in the other. You could also override this to
13 1211 --     create a read-only copy of the document. )
13 1212 --
13 1213 -- PROCEDURE TDocument.ReadFromFile(VAR anAppFile: AppFile;
13 1214 --   forPrinting: BOOLEAN);
13 1215 --   { Called to read an existing file for display or printing. If
13 1216 --     anAppFile.fName = '' THEN we read from the file specified in
13 1217 --     the instance variables instead of the parameter, and the
13 1218 --     parameter will be updated to match the document. }
13 1219 --
13 1220 -- PROCEDURE TDocument.ShowWindows;
13 1221 --   { Called when opening the document initially; default is to call
13 1222 --     OpenWindow for all windows that have fFlags.openInitially TRUE }
13 1223 --
13 1224 -- PROCEDURE TDocument.UntitledName(VAR noName: Str255);
13 1225 --   { Called to supply a name for an untitled document. If this returns
13 1226 --     '' then the windows are not renamed from the value specified
13 1227 --     in their titles. }
13 1228 --
13 1229 --
13 1230 -- { Saving Documents }
13 1231 -- PROCEDURE TDocument.AboutToSave(itsCmd: CmdNumber; VAR newName: Str255;
13 1232 --   VAR newVolRefnum: INTEGER; VAR makingCopy: BOOLEAN);
13 1233 --   { Called right after the SFPutFile (if any) and before deciding
13 1234 --     between SaveViaTemp & SaveInPlace. Applications can use the
13 1235 --     name in an alert and/or modify the makingCopy parameter if they
13 1236 --     do not want to have the document renamed. }
13 1237 --
13 1238 -- PROCEDURE TDocument.DoNeedDiskSpace(VAR dataForkBytes,
13 1239 --   rsrcForkBytes: LONGINT);
13 1240 --   { Bytes required to store SAVE file(s) on disk. When called, both
13 1241 --     parameters are set to 0 for you. }
13 1242 --
13 1243 -- PROCEDURE TDocument.FreeFile;
13 1244 --   { Called to free in memory data associated with the currently opened
13 1245 --     document. The default is to call CloseFile(fDataRefnum,
13 1246 --     fRsrcRefnum) if fDataOpen OR fRsrcOpen. }
13 1247 --
13 1248 -- FUNCTION TDocument.GetSaveInfo(itsCmdNumber: CmdNumber;
13 1249 --   copyFInfo: BOOLEAN; VAR cInfo: CInfoPBRc): BOOLEAN;
13 1250 --   { Called to get the desired file info for saving the file. Default
13 1251 --     copies the info from the current file if fSaveExists AND
13 1252 --     copyFInfo, otherwise it just sets the file type and creator
13 1253 --     based on the fields of SELF. If an error occurs calling the
13 1254 --     file system, the cInfo record will not be filled in. Returns
13 1255 --     TRUE if it successfully gets the info from the file. }
13 1256 --
13 1257 -- PROCEDURE TDocument.GetTempName(VAR filename: Str255);
13 1258 --   { Return a temporary name for saving the file. The filename is
13 1259 --     constructed in two parts: the first part is the document's
13 1260 --     name, or the application's name if the document is untitled.
13 1261 --     The second part, appended to the first part, is a pseudo-
13 1262 --     random number based on the tick count and number of seconds
13 1263 --     since boot. }
13 1264 --
13 1265 -- PROCEDURE TDocument.MakeNewCopy(makingCopy: BOOLEAN;
13 1266 --   validFInfo: BOOLEAN; VAR cInfo: CInfoPBRc);
13 1267 --   { Creates a brand new copy of the document in the file specified by
13 1268 --     cInfo. The name, volume refnum, file type, and creator must
13 1269 --     be specified in cInfo. The other info fields of cInfo are used
13 1270 --     only if makingCopy is FALSE.
13 1271 --
13 1272 --     makingCopy is TRUE if this save is making a copy of the
13 1273 --     document (vs. a regular Save or Save As...)
13 1274 --
13 1275 --     Caller is responsible for calling FlushVol after this. }
13 1276 --
13 1277 -- FUNCTION TDocument.PoseSaveDialog: INTEGER;
13 1278 --   { If the document has been changed (fChangeCount > 0), then the
13 1279 --     user is asked to save the document. PoseSaveDialog returns
13 1280 --     cancel, kYesButton or kNoButton, depending on the button
13 1281 --     chosen by the user. If the document has not been changed then
13 1282 --     kNoButton is returned. }
13 1283 --
13 1284 -- PROCEDURE TDocument.RequestFileName (itsCmdNumber: CmdNumber;
13 1285 --   makingCopy: BOOLEAN;
13 1286 --   VAR filename: Str255;

```

```

13 1287 --                                VAR volRefNum: INTEGER);
13 1288 --                                { Asks the user for a filename for the document. Calls
13 1289 --                                TDocument.SFPutParms and then SFPutFile. Fails if the user
13 1290 --                                clicks the Cancel button to end the put file dialog. }
13 1291 --
13 1292 --                                PROCEDURE TDocument.Save(itsCmdNumber: CmdNumber;
13 1293 --                                askForFilename, makingCopy: BOOLEAN);
13 1294 --                                { Try to save the document to disk; If askForFilename, put up a Std
13 1295 --                                File box; If makingCopy, then this is for the Save a Copy In
13 1296 --                                command, so we do not rename the document.
13 1297 --
13 1298 --                                This method calls FlushVol (which takes care of the requirements
13 1299 --                                of MakeNewCopy, SaveInPlace, and SaveViaTemp. )
13 1300 --
13 1301 --                                PROCEDURE TDocument.SaveAgain(itsCmdNumber: CmdNumber;
13 1302 --                                makingCopy: BOOLEAN; savingDoc: TDocument);
13 1303 --                                { Called when the user tries to Save As_ or Save a Copy In_
13 1304 --                                and specifies the name of a document that is already opened.
13 1305 --                                The default case calls Failure(errSaveAgain, msgSaveFailed) to
13 1306 --                                display an error. (The goal is to prevent the user from
13 1307 --                                having 2 copies of the same document open, in which case s/he
13 1308 --                                might make incompatible changes to both copies.)
13 1309 --
13 1310 --                                savingDoc is the document being saved )
13 1311 --
13 1312 --                                PROCEDURE TDocument.SavedOn(VAR fileName: Str255; volRefnum: INTEGER);
13 1313 --                                { Called when a save successfully completes and we are switching
13 1314 --                                to the new file (ie., not a Save a Copy In_). }
13 1315 --
13 1316 --                                PROCEDURE TDocument.SaveInPlace(itsCmdNumber: CmdNumber;
13 1317 --                                makingCopy, copyFInfo: BOOLEAN; VAR fileName: Str255;
13 1318 --                                volRefnum: INTEGER);
13 1319 --                                { Called to save a document in place. Default is to delete the
13 1320 --                                existing file if NOT (fDataOpen OR fRsrcOpen). If you have
13 1321 --                                disk based document, you have to decide whether to allow saving
13 1322 --                                in place.
13 1323 --
13 1324 --                                makingCopy is TRUE if this save is making a copy of the document
13 1325 --                                (vs. a regular Save or Save As_)
13 1326 --
13 1327 --                                copyFInfo is TRUE if the file information should be copied to the
13 1328 --                                new file
13 1329 --
13 1330 --                                Caller is responsible for calling FlushVol after this. }
13 1331 --
13 1332 --                                PROCEDURE TDocument.SaveViaTemp(itsCmdNumber: CmdNumber;
13 1333 --                                makingCopy, copyFInfo: BOOLEAN; VAR fileName: Str255;
13 1334 --                                volRefnum: INTEGER);
13 1335 --                                { Called to save the document by creating a temporary copy of the
13 1336 --                                file and renaming it. MacApp uses this method except when
13 1337 --                                there is not enough disk space available.
13 1338 --
13 1339 --                                makingCopy is TRUE if this save is making a copy of the document
13 1340 --                                (vs. a regular Save or Save As_)
13 1341 --
13 1342 --                                copyFInfo is TRUE if the file information should be copied to the
13 1343 --                                new file
13 1344 --
13 1345 --                                Caller is responsible for calling FlushVol after this. }
13 1346 --
13 1347 --                                PROCEDURE TDocument.SFPutParms(itsCmdNumber: CmdNumber;
13 1348 --                                VAR dlgID: INTEGER; VAR where: Point;
13 1349 --                                VAR defaultName, prompt: Str255;
13 1350 --                                VAR dlgHook, filterProc: ProcPtr);
13 1351 --                                { Called to return all the parameters that should be passed to
13 1352 --                                SFPutFile.
13 1353 --                                Defaults to
13 1354 --                                dlgID = putDlgID;
13 1355 --                                where = (100, 100);
13 1356 --                                defaultName not changed;
13 1357 --                                prompt gotten from resource file;
13 1358 --                                dlgHook = NIL;
13 1359 --                                filterProc = NIL. }
13 1360 --
13 1361 --
13 1362 --                                { Closing Documents }
13 1363 --                                PROCEDURE TDocument.Close;
13 1364 --                                { Close a document and free it. Signals Failure with error = noErr
13 1365 --                                and message = msgCancelled if user cancels. }
13 1366 --                                { ??? NOTE: Must never be called for a document related to a view
13 1367 --                                in the Clipboard. }
13 1368 --
13 1369 --
13 1370 --
13 1371 --                                { Revert }
13 1372 --                                PROCEDURE TDocument.Revert;
13 1373 --                                PROCEDURE TDocument.ShowReverted;
13 1374 --                                { Called after Revert to show brand new document; default tells each
13 1375 --                                view to ShowReverted }
13 1376 --
13 1377 --
13 1378 --                                { Miscellaneous }
13 1379 --                                PROCEDURE TDocument.CheckDiskFile(rsrcID, rsrcIndex: INTEGER;
13 1380 --                                reverting: BOOLEAN);
13 1381 --                                { Check to see if the disk file has changed. If so put up alert
13 1382 --                                phFileChanged after setting ^0 to fTitle and ^1 to the string
13 1383 --                                specified by the rsrcID and rsrcIndex. If user cancels, signal
13 1384 --                                Failure(noErr, msgCancelled).
13 1385 --

```

```

13 1386 --      If reverting is TRUE, then I/O errors and nonmatching file types
13 1387 --      will cause a Failure.
13 1388 --      Otherwise, this is a Save and any I/O errors and nonmatching
13 1389 --      file type will be ignored. }
13 1390 --
13 1391 --      FUNCTION TDocument.DiskFileChanged(checkType: BOOLEAN): OSErr;
13 1392 --      { Returns noErr if fSaveExists is FALSE or if the file's
13 1393 --      modification date matches fModDate. If checkType is TRUE then
13 1394 --      returns errTypeChanged if the file type has changed. If the
13 1395 --      file exists but with a different modification date, returns
13 1396 --      errFileChanged. }
13 1397 --
13 1398 --      PROCEDURE TDocument.SetTitle(aTitle: Str255);
13 1399 --      { Sets SELF.fTitle to aTitle and calls SetTitleForDoc for each
13 1400 --      window of the document. }
13 1401 --
13 1402 --
13 1403 --      { Command Handlers }
13 1404 --      FUNCTION TDocument.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
13 1405 --      PROCEDURE TDocument.DoSetupMenus; OVERRIDE;
13 1406 --
13 1407 --
13 1408 --      { Hierarchy }
13 1409 --      PROCEDURE TDocument.AddView(aView: TView);
13 1410 --      { Adds the view to the end of the document's view list. }
13 1411 --
13 1412 --      PROCEDURE TDocument.AddWindow(aWindow: TWindow);
13 1413 --      { Adds the window to the end of the document's window list. }
13 1414 --
13 1415 --      PROCEDURE TDocument.DeleteView(viewToDelete: TView);
13 1416 --      { Deletes the view from the document's view list. }
13 1417 --
13 1418 --      PROCEDURE TDocument.DeleteWindow(windowToDelete: TWindow);
13 1419 --      { Deletes the window from the document's window list. }
13 1420 --
13 1421 --      PROCEDURE TDocument.ForAllViewsDo(PROCEDURE DoToView(aView: TView));
13 1422 --      { Performs DoToView on every view in every frame of every window
13 1423 --      belonging to the document. If you want to deal with all views
13 1424 --      belonging to the document use fViewList.Each. }
13 1425 --
13 1426 --      PROCEDURE TDocument.ForAllWindowsDo(PROCEDURE DoToWind(aWindow: TWindow));
13 1427 --      { Perform the supplied proc to each window belonging to the document }
13 1428 --
13 1429 --      END;
13 1430 --
13 1431 --
13 1432 --      {-----+
13 1433 --      | TView                                     |
13 1434 --      +-----+}
13 1435 --      ViewTemplatePtr = ^ViewTemplate;
13 1436 --      ViewTemplate = PACKED RECORD
13 1437 --      itsParentID: IDType;
13 1438 --      thisViewID: IDType;
13 1439 --      itsLocation: VPoint;
13 1440 --      itsSize: VPoint;
13 1441 --      itsVSizeDet: Integer8; { Coerced into a SizeDeterminer }
13 1442 --      itsHSizeDet: Integer8; { Coerced into a SizeDeterminer }
13 1443 --      isShown: BOOLEAN;
13 1444 --      isEnabled: BOOLEAN;
13 1445 --      filler: Byte;
13 1446 --      itsSignature: IDType; { Resource signature }
13 1447 --      CASE INTEGER OF
13 1448 --          0: (itsType: Str255); { Actually variable length }
13 1449 --          1: (includeRsrcID: INTEGER);
13 1450 --      END;
13 1451 --
13 1452 --      ViewRsrcHndl = ^ViewRsrcPtr;
13 1453 --      ViewRsrcPtr = ^ViewResource;
13 1454 --      ViewResource = RECORD
13 1455 --      numViews: INTEGER;
13 1456 --      theViews: ViewTemplate; { Actually, open-ended }
13 1457 --      END;
13 1458 --
13 1459 --      TView = OBJECT (TEvtHandler)
13 1460 --
13 1461 --      fSuperView: TView;
13 1462 --      fSubViews: TList;
13 1463 --      fDocument: TDocument;
13 1464 --      fLocation: VPoint;
13 1465 --      fSize: VPoint;
13 1466 --      fSizeDeterminer: ARRAY [VHSelect] OF SizeDeterminer;
13 1467 --      fHLDesired: HLState;
13 1468 --      fIdentifier: IDType;
13 1469 --      fShown: BOOLEAN;
13 1470 --      fViewEnabled: BOOLEAN;
13 1471 --      fPrintHandler: TPrintHandler;
13 1472 --
13 1473 --      { Creation/Destruction Methods }
13 1474 --
13 1475 --      PROCEDURE TView.IView (itsDocument: TDocument;
13 1476 --      itsSuperview: TView;
13 1477 --      itsLocation: VPoint;
13 1478 --      itsSize: VPoint;
13 1479 --      itsHSizeDet,
13 1480 --      itsVSizeDet: SizeDeterminer);
13 1481 --
13 1482 --      PROCEDURE TView.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr);
13 1483 --
13 1484 --      PROCEDURE TView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr);

```

```

13 1485 --
13 1486 --      PROCEDURE TView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr);
13 1487 --
13 1488 --      PROCEDURE TView.Free; OVERRIDE;
13 1489 --
13 1490 --      PROCEDURE TView.FreeFromClipboard;
13 1491 --
13 1492 --      ( Coordinate Conversion Methods )
13 1493 --
13 1494 --      PROCEDURE TView.QDToViewPt (qdPoint: Point; VAR viewPt: VPoint);
13 1495 --
13 1496 --      PROCEDURE TView.QDToViewRect (qdRect: Rect; VAR viewRect: VRect);
13 1497 --
13 1498 --      FUNCTION TView.ViewToQDPt (viewPt: VPoint): Point;
13 1499 --
13 1500 --      PROCEDURE TView.ViewToQDRect (viewRect: VRect; VAR qdRect: Rect);
13 1501 --
13 1502 --      PROCEDURE TView.LocalToSuper (VAR thePoint: VPoint);
13 1503 --          { Given a point in local view coordinates, returns the same
13 1504 --            point in its superview's coordinate system. }
13 1505 --
13 1506 --      PROCEDURE TView.SuperToLocal (VAR thePoint: VPoint);
13 1507 --          { Given a point in superview coordinates, returns the same point
13 1508 --            in the view's own coordinate system. }
13 1509 --
13 1510 --      PROCEDURE TView.LocalToWindow (VAR thePoint: VPoint);
13 1511 --          { Given a point in local view coordinates, returns the same
13 1512 --            point in its window's (or most super superview's) coordinate system. }
13 1513 --
13 1514 --      PROCEDURE TView.WindowToLocal (VAR thePoint: VPoint);
13 1515 --          { Given a point in window (or most super superview) coordinates, returns
13 1516 --            the same point in the view's own coordinate system. }
13 1517 --
13 1518 --      ( Subview Management Methods )
13 1519 --
13 1520 --      PROCEDURE TView.EachSubview (PROCEDURE DoToSubview (theSubview: TView));
13 1521 --
13 1522 --      FUNCTION TView.FirstSubviewThat (
13 1523 --          FUNCTION TestSubview (theSubview: TView): BOOLEAN): TView;
13 1524 --
13 1525 --      FUNCTION TView.LastSubviewThat (
13 1526 --          FUNCTION TestSubview (theSubview: TView): BOOLEAN): TView;
13 1527 --
13 1528 --      PROCEDURE TView.AddSubview (theSubview: TView);
13 1529 --
13 1530 --      PROCEDURE TView.RemoveSubview (theSubview: TView);
13 1531 --
13 1532 --      PROCEDURE TView.MakeFirstSubview (theSubview: TView);
13 1533 --
13 1534 --      PROCEDURE TView.MakeLastSubview (theSubview: TView);
13 1535 --
13 1536 --      FUNCTION TView.CountSubviews: INTEGER;
13 1537 --
13 1538 --      FUNCTION TView.FindSubview (itsIdentifier: IDType): TView;
13 1539 --
13 1540 --      ( Open/Close/Activate Methods )
13 1541 --
13 1542 --      PROCEDURE TView.Activate (entering: BOOLEAN);
13 1543 --
13 1544 --      PROCEDURE TView.Open;
13 1545 --
13 1546 --      PROCEDURE TView.Close;
13 1547 --
13 1548 --      PROCEDURE TView.BeInScroller (itsScroller: TScroller);
13 1549 --
13 1550 --      PROCEDURE TView.BeInPort (itsPort: GrafPtr);
13 1551 --
13 1552 --      PROCEDURE TView.ShowReverted;
13 1553 --
13 1554 --      ( Choice Handling Methods )
13 1555 --
13 1556 --      PROCEDURE TView.DoChoice (origView: TView; itsChoice: INTEGER);
13 1557 --
13 1558 --      ( Size Methods )
13 1559 --
13 1560 --      PROCEDURE TView.AdjustSize;
13 1561 --
13 1562 --      PROCEDURE TView.CalcMinSize (VAR minSize: VPoint);
13 1563 --
13 1564 --      PROCEDURE TView.ComputeSize (VAR newSize: VPoint);
13 1565 --
13 1566 --      PROCEDURE TView.Resize (width, height: VCoordinate; invalidate: BOOLEAN);
13 1567 --
13 1568 --      PROCEDURE TView.SuperViewChangedSize (delta: VPoint; invalidate: BOOLEAN);
13 1569 --
13 1570 --      PROCEDURE TView.SubViewChangedSize (theSubview: TView; delta: VPoint);
13 1571 --
13 1572 --      ( Location Methods )
13 1573 --
13 1574 --      PROCEDURE TView.Locate (h, v: VCoordinate; invalidate: BOOLEAN);
13 1575 --
13 1576 --      PROCEDURE TView.SubViewMoved (theSubview: TView);
13 1577 --
13 1578 --      PROCEDURE TView.SuperViewMoved (invalidate: BOOLEAN);
13 1579 --
13 1580 --      ( Focusing Methods )
13 1581 --
13 1582 --      FUNCTION TView.Focus: BOOLEAN;
13 1583 --

```

```

13 1584 --      FUNCTION TView.FocusOnSuperview: BOOLEAN;
13 1585 --
13 1586 --      PROCEDURE TView.ClipFurtherTo(r: Rect; hDeltaOrg, vDeltaOrg: INTEGER);
13 1587 --
13 1588 --      ( Drawing Methods )
13 1589 --
13 1590 --      PROCEDURE TView.DrawContents;
13 1591 --
13 1592 --      PROCEDURE TView.Draw (area: Rect);
13 1593 --
13 1594 --      PROCEDURE TView.DoHighlightSelection (fromHL, toHL: HLState);
13 1595 --
13 1596 --      PROCEDURE TView.Adorn (area: Rect; itsPenSize: Point; itsAdornment: CntlAdornment);
13 1597 --
13 1598 --      PROCEDURE TView.Update;
13 1599 --
13 1600 --      ( Validation/Invalidation Methods )
13 1601 --
13 1602 --      PROCEDURE TView.ForceRedraw;
13 1603 --
13 1604 --      PROCEDURE TView.InvalidVRect (viewRect: VRect);
13 1605 --
13 1606 --      PROCEDURE TView.ValidVRect (viewRect: VRect);
13 1607 --
13 1608 --      PROCEDURE TView.InvalidRect (r: Rect);
13 1609 --
13 1610 --      PROCEDURE TView.RevealRect (rectToReveal: VRect; minToSee: Point;
13 1611 --                                redraw: BOOLEAN);
13 1612 --
13 1613 --      PROCEDURE TView.RevealTop (redraw: BOOLEAN);
13 1614 --
13 1615 --      PROCEDURE TView.RevealBottom (redraw: BOOLEAN);
13 1616 --
13 1617 --      ( Mouse Handling Methods )
13 1618 --
13 1619 --      FUNCTION TView.ContainsMouse (theMouse: VPoint): BOOLEAN;
13 1620 --
13 1621 --      FUNCTION TView.HandleMouseDown (theMouse: VPoint; VAR info: EventInfo;
13 1622 --                                     VAR hysteresis: Point;
13 1623 --                                     VAR theCommand: TCommand): BOOLEAN;
13 1624 --
13 1625 --      FUNCTION TView.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
13 1626 --                                     VAR hysteresis: Point): TCommand;
13 1627 --
13 1628 --      ( Cursor Handling Methods )
13 1629 --
13 1630 --      FUNCTION TView.HandleCursor (theMouse: VPoint; cursorRgn: RgnHandle): TView;
13 1631 --
13 1632 --      FUNCTION TView.DoSetCursor (localPoint: Point;
13 1633 --                                cursorRgn: RgnHandle): BOOLEAN;
13 1634 --
13 1635 --      ( Miscellaneous Methods )
13 1636 --
13 1637 --      FUNCTION TView.GetDialogView: TView; ( actually, TDialogView_ )
13 1638 --
13 1639 --      FUNCTION TView.GetGrafPort: GrafPtr;
13 1640 --
13 1641 --      FUNCTION TView.GetScroller (immediateSuperview: BOOLEAN): TScroller;
13 1642 --
13 1643 --      FUNCTION TView.GetWindow: TWindow;
13 1644 --
13 1645 --      PROCEDURE TView.GetExtent (VAR itsExtent: VRect);
13 1646 --
13 1647 --      PROCEDURE TView.GetQDExtent (VAR qdExtent: Rect);
13 1648 --
13 1649 --      PROCEDURE TView.GetFrame (VAR itsFrame: VRect);
13 1650 --
13 1651 --      PROCEDURE TView.GetVisibleRect (VAR visQDRect: Rect);
13 1652 --
13 1653 --      PROCEDURE TView.ViewEnable (state, redraw: BOOLEAN);
13 1654 --
13 1655 --      FUNCTION TView.IsViewEnabled: BOOLEAN;
13 1656 --
13 1657 --      PROCEDURE TView.Show (state, redraw: BOOLEAN);
13 1658 --
13 1659 --      FUNCTION TView.IsShown: BOOLEAN;
13 1660 --
13 1661 --      ( Clipboard Handling )
13 1662 --
13 1663 --      FUNCTION TView.ContainsClipType (aType: ResType): BOOLEAN;
13 1664 --
13 1665 --      FUNCTION TView.GivePasteData (aDataHandle: Handle; dataType: ResType): LONGINT;
13 1666 --
13 1667 --      PROCEDURE TView.WriteToDeskScrap;
13 1668 --
13 1669 --      ( PrintingMethods )
13 1670 --
13 1671 --      PROCEDURE TView.AttachPrintHandler(itsPrintHandler: TPrintHandler);
13 1672 --      ( Called by the PrintHandler during its initialization, NOT directly
13 1673 --        by you )
13 1674 --
13 1675 --      FUNCTION TView.DoBreakFollowing(vhs: VHSelect; prevBreak: VCoordinate;
13 1676 --                                     VAR automatic: BOOLEAN): VCoordinate;
13 1677 --      ( Returns the location of the page break which follows the page break
13 1678 --        located at 'prevBreak', in direction vhs; returns automatic =
13 1679 --        TRUE if the page-break is thought of as an 'automatic' rather
13 1680 --        than a 'manual' (user-specified) one. Note that page-breaks in
13 1681 --        dimension 'v' are drawn as vertical lines, those in dimension
13 1682 --        'h' as horizontal lines )

```

```

13 1683 --
13 1684 --      PROCEDURE TView.DoCalcPageStrips (VAR pageStrips: Point);
13 1685 --      { Computes the number of horizontal and vertical page strips. }
13 1686 --
13 1687 --      PROCEDURE TView.DoCalcViewPerPage (VAR viewPerPage: VPoint);
13 1688 --      { Determine how much of the view normally goes into each printed
13 1689 --        page, in each dimension. }
13 1690 --
13 1691 --      PROCEDURE TView.DoCheckPrinter;
13 1692 --      { Check whether printer (if relevant) has changed, and if so, take
13 1693 --        appropriate action }
13 1694 --
13 1695 --      PROCEDURE TView.DoDrawPrintFeedback(area: Rect);
13 1696 --      { Draw page-breaks, page-numbers, view-borders, rulers, etc. }
13 1697 --
13 1698 --      PROCEDURE TView.DoDrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
13 1699 --        loc: VCoordinate; automatic: BOOLEAN);
13 1700 --      { Draw one page break. }
13 1701 --
13 1702 --      FUNCTION TView.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
13 1703 --      { Enables my fPrintHandler to handle commands }
13 1704 --
13 1705 --      PROCEDURE TView.DoPagination;
13 1706 --      { Recompute the dividing lines between areas of the view which will
13 1707 --        be mapped into different printed pages }
13 1708 --
13 1709 --      PROCEDURE TView.DoPrinterChanged;
13 1710 --      { The metrics relating to printer use have changed; absorb the
13 1711 --        information and react }
13 1712 --
13 1713 --      PROCEDURE TView.DoSetPageOffset(coord: VPoint);
13 1714 --      { Only reimplement in very unusual circumstances involving
13 1715 --        Tall-UnAdjusted text printing etc. }
13 1716 --
13 1717 --      PROCEDURE TView.DoSetupMenus; OVERRIDE;
13 1718 --      { tells my PrintHandler to set up its menus }
13 1719 --
13 1720 --      PROCEDURE TView.GetPrintExtent (VAR printExtent: VRect);
13 1721 --      { Returns the part of the view that is to be printed. The default is
13 1722 --        to return the view's extent. }
13 1723 --
13 1724 --      PROCEDURE TView.PageInteriorChanged(newInterior: Rect);
13 1725 --      { Allows a view which is printable to react to the changing of the
13 1726 --        page-interior rectangle }
13 1727 --
13 1728 --      { Debugging Methods }
13 1729 --
13 1730 --      {$IFC qDebug}
13 1731 --      PROCEDURE TView.Fields (PROCEDURE DoToField (fieldName: Str255;
13 1732 --        fieldAddr: Ptr;
13 1733 --        fieldType: INTEGER)); OVERRIDE;
13 1734 --
13 1735 --      PROCEDURE TView.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
13 1736 --      {$ENDC}
13 1737 --
13 1738 --      END ( TView );
13 1739 --
13 1740 --      {-----+
13 1741 --      | TScroller |
13 1742 --      +-----}
13 1743 --      ScrollerTemplatePtr = ^ScrollerTemplate;
13 1744 --      ScrollerTemplate = RECORD
13 1745 --          wantVSBBar:    BOOLEAN;
13 1746 --          wantHSBar:    BOOLEAN;
13 1747 --          vertMax:      LONGINT;
13 1748 --          horzMax:      LONGINT;
13 1749 --          vScrollUnits: INTEGER;
13 1750 --          hScrollUnits: INTEGER;
13 1751 --          vConstrain:   BOOLEAN;
13 1752 --          hConstrain:   BOOLEAN;
13 1753 --          sBarOffsets:  Rect;
13 1754 --      END;
13 1755 --
13 1756 --      TScroller = OBJECT (TView)
13 1757 --
13 1758 --          fTranslation:    VPoint;
13 1759 --          fScrollLimit:    VPoint;
13 1760 --          fMaxTranslation: VPoint;
13 1761 --          fScrollBars:     ARRAY [VHSelect] OF TSScrollBar;
13 1762 --          fScrollUnit:     Point;
13 1763 --          fConstrain:      ARRAY [VHSelect] OF BOOLEAN;
13 1764 --          fSBarOffsets:    VRect;
13 1765 --
13 1766 --      { Creation/Destruction Methods }
13 1767 --
13 1768 --      {$IFC qProceduralViews}
13 1769 --      PROCEDURE TScroller.IScroller (itsSuperView: TView; itsLocation, itsSize: VPoint;
13 1770 --        itsHSizeDet, itsVSizeDet: SizeDeterminer;
13 1771 --        itsHorzMax, itsVertMax: VCoordinate;
13 1772 --        wantHorzSBar, wantVertSBar: BOOLEAN);
13 1773 --      {$ENDC}
13 1774 --
13 1775 --      {$IFC qTemplateViews}
13 1776 --      PROCEDURE TScroller.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
13 1777 --      {$ENDC}
13 1778 --
13 1779 --      {$IFC qWriteTemplates}
13 1780 --      PROCEDURE TScroller.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 1781 --

```

```

13 1782 --      PROCEDURE TScroller.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 1783 --      ($ENDC)
13 1784 --
13 1785 --      PROCEDURE TScroller.Free; OVERRIDE;
13 1786 --
13 1787 --      ( Subview Management Methods )
13 1788 --
13 1789 --      PROCEDURE TScroller.AddSubView (theSubView: TView); OVERRIDE;
13 1790 --
13 1791 --      PROCEDURE TScroller.RemoveSubView (theSubView: TView); OVERRIDE;
13 1792 --
13 1793 --      ( Focusing Methods )
13 1794 --
13 1795 --      FUNCTION TScroller.Focus: BOOLEAN; OVERRIDE;
13 1796 --
13 1797 --      ( Miscellaneous Methods )
13 1798 --
13 1799 --      PROCEDURE TScroller.LocalToSuper (VAR thePoint: VPoint); OVERRIDE;
13 1800 --
13 1801 --      PROCEDURE TScroller.SuperToLocal (VAR thePoint: VPoint); OVERRIDE;
13 1802 --      { Given a point in superview coordinates, returns the same point
13 1803 --        in the view's own coordinate system. }
13 1804 --
13 1805 --      PROCEDURE TScroller.GetExtent (VAR itsExtent: VRect); OVERRIDE;
13 1806 --
13 1807 --      ( Size Methods )
13 1808 --
13 1809 --      PROCEDURE TScroller.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 1810 --
13 1811 --      ( Location Methods )
13 1812 --
13 1813 --      PROCEDURE TScroller.Locate (h, v: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 1814 --
13 1815 --      PROCEDURE TScroller.AdjustScrollBars;
13 1816 --
13 1817 --      PROCEDURE TScroller.SubViewChangedSize (theSubView: TView; delta: VPoint); OVERRIDE;
13 1818 --
13 1819 --      PROCEDURE TScroller.SetScrollLimits (scrollLimit: VPoint; drawScrollBars: BOOLEAN);
13 1820 --
13 1821 --      PROCEDURE TScroller.SetScrollParameters (horzUnits, vertUnits: VCoordinate;
13 1822 --        horzConstraint, vertConstraint: BOOLEAN);
13 1823 --
13 1824 --      ( Scrolling Methods )
13 1825 --
13 1826 --      PROCEDURE TScroller.CreateScrollBars (wantHorzSBar, wantVertSBar: BOOLEAN);
13 1827 --
13 1828 --      PROCEDURE TScroller.HaveScrollBar (theScrollBar: TSScrollBar;
13 1829 --        direction: VHSelect);
13 1830 --
13 1831 --      PROCEDURE TScroller.ScrollDraw (delta: VPoint);
13 1832 --
13 1833 --      PROCEDURE TScroller.DoScroll (delta: VPoint; redraw: BOOLEAN);
13 1834 --
13 1835 --      FUNCTION TScroller.ScrollStep (vhs: VHSelect; partCode: INTEGER): VCoordinate;
13 1836 --
13 1837 --      FUNCTION TScroller.ScrollRelative (vhs: VHSelect; sBarValue: VCoordinate): VCoordinate;
13 1838 --
13 1839 --      PROCEDURE TScroller.ScrollBy (deltaH, deltaV: VCoordinate; redraw: BOOLEAN);
13 1840 --
13 1841 --      PROCEDURE TScroller.ScrollTo (h, v: VCoordinate; redraw: BOOLEAN);
13 1842 --
13 1843 --      PROCEDURE TScroller.AutoScroll (viewPt: VPoint; VAR delta: VPoint);
13 1844 --
13 1845 --      PROCEDURE TScroller.RevealRect (rectToReveal: VRect; minToSee: Point;
13 1846 --        redraw: BOOLEAN); OVERRIDE;
13 1847 --
13 1848 --      ( Miscellaneous Methods )
13 1849 --
13 1850 --      FUNCTION TScroller.GetScroller (immediateSuperView: BOOLEAN): TScroller; OVERRIDE;
13 1851 --
13 1852 --      ( Debugging Methods )
13 1853 --
13 1854 --      ($IFC qDebug)
13 1855 --      PROCEDURE TScroller.Fields (PROCEDURE DoToField (fieldName: Str255;
13 1856 --        fieldAddr: Ptr;
13 1857 --        fieldType: INTEGER)); OVERRIDE;
13 1858 --      ($ENDC)
13 1859 --
13 1860 --      END { TScroller };
13 1861 --
13 1862 --      (-----+
13 1863 --      | TWindow |
13 1864 --      +-----+
13 1865 --      WindowTemplatePtr = ^WindowTemplate;
13 1866 --      WindowTemplate = PACKED RECORD
13 1867 --          procID: INTEGER;
13 1868 --          hasGoAway: BOOLEAN;
13 1869 --          resizable: BOOLEAN;
13 1870 --          isModal: BOOLEAN;
13 1871 --          doFirstClick: BOOLEAN;
13 1872 --          freeOnClosing: BOOLEAN;
13 1873 --          disposeOnFree: BOOLEAN;
13 1874 --          closesDocument: BOOLEAN;
13 1875 --          openInitially: BOOLEAN;
13 1876 --          mustAdaptToScreen: BOOLEAN;
13 1877 --          stagger: BOOLEAN;
13 1878 --          mustForceOnScreen: BOOLEAN;
13 1879 --          vertCenter: BOOLEAN;
13 1880 --          horzCenter: BOOLEAN;

```



```

13 1881 --          filler:          0..7;
13 1882 --          targetID:       IDType;
13 1883 --          title:          Str255; (Actually variable length)
13 1884 --          END;
13 1885 --
13 1886 --      WindowFlags =      PACKED RECORD
13 1887 --                          adapted:      BOOLEAN;
13 1888 --                          hCentered:    BOOLEAN;
13 1889 --                          vCentered:    BOOLEAN;
13 1890 --                          staggered:    BOOLEAN;
13 1891 --                          forced:       BOOLEAN;
13 1892 --                          isActive:     BOOLEAN;
13 1893 --                          isResizable:   BOOLEAN;
13 1894 --                          isClosable:    BOOLEAN;
13 1895 --                          freeOnClosing: BOOLEAN;
13 1896 --                          disposeOnFree: BOOLEAN;
13 1897 --                          closesDocument: BOOLEAN;
13 1898 --                          openInitially: BOOLEAN;
13 1899 --                          isModal:       BOOLEAN;
13 1900 --                          doFirstClick:  BOOLEAN;
13 1901 --          END;
13 1902 --
13 1903 --      TWindow =          OBJECT (TView)
13 1904 --
13 1905 --          fWmgrWindow:    WindowPtr;
13 1906 --          fProcID:        INTEGER;
13 1907 --          fMoveBounds:    Rect;
13 1908 --          fResizeLimits:  Rect;
13 1909 --          fTarget:        TEvtHandler;
13 1910 --          fTargetID:      IDType;
13 1911 --          fPreDocname:    INTEGER;
13 1912 --          fConstTitle:    INTEGER;
13 1913 --          fIsActive:      BOOLEAN;
13 1914 --          fIsResizable:    BOOLEAN;
13 1915 --          fIsClosable:     BOOLEAN;
13 1916 --          fFreeOnClosing:  BOOLEAN;
13 1917 --          fDisposeOnFree:  BOOLEAN;
13 1918 --          fClosesDocument: BOOLEAN;
13 1919 --          fOpenInitially:  BOOLEAN;
13 1920 --          fIsModal:        BOOLEAN;
13 1921 --          fDoFirstClick:   BOOLEAN;
13 1922 --          fAdapted:        BOOLEAN;
13 1923 --          fHorzCentered:   BOOLEAN;
13 1924 --          fVertCentered:   BOOLEAN;
13 1925 --          fStaggered:      BOOLEAN;
13 1926 --          fForcedOnScreen:  BOOLEAN;
13 1927 --
13 1928 --          { Creation/Destruction Methods }
13 1929 --
13 1930 --      ($IFC qProceduralViews)
13 1931 --          PROCEDURE TWindow.IWindow (itsDocument: TDocument; itsWmgrWindow: WindowPtr;
13 1932 --                                     canResize, canClose, disposeOnFree: BOOLEAN);
13 1933 --      ($ENDC)
13 1934 --
13 1935 --      ($IFC qTemplateViews)
13 1936 --          PROCEDURE TWindow.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
13 1937 --      ($ENDC)
13 1938 --
13 1939 --      ($IFC qWriteTemplates)
13 1940 --          PROCEDURE TWindow.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 1941 --
13 1942 --          PROCEDURE TWindow.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 1943 --      ($ENDC)
13 1944 --
13 1945 --          PROCEDURE TWindow.Free; OVERRIDE;
13 1946 --
13 1947 --          { Open/Close Methods }
13 1948 --
13 1949 --          PROCEDURE TWindow.Open; OVERRIDE;
13 1950 --
13 1951 --          PROCEDURE TWindow.Close; OVERRIDE;
13 1952 --
13 1953 --          PROCEDURE TWindow.CloseByUser;
13 1954 --
13 1955 --          { Activation Methods }
13 1956 --
13 1957 --          PROCEDURE TWindow.Activate (entering: BOOLEAN); OVERRIDE;
13 1958 --
13 1959 --          { Size/Location Methods }
13 1960 --
13 1961 --          PROCEDURE TWindow.Locate (h, v: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 1962 --
13 1963 --          PROCEDURE TWindow.MoveByUser (globalMouse: Point);
13 1964 --
13 1965 --          PROCEDURE TWindow.ResizeByUser (globalMouse: Point);
13 1966 --
13 1967 --          PROCEDURE TWindow.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 1968 --
13 1969 --          PROCEDURE TWindow.Zoom (partCode: INTEGER);
13 1970 --
13 1971 --          { Focusing Methods }
13 1972 --
13 1973 --          FUNCTION TWindow.Focus: BOOLEAN; OVERRIDE;
13 1974 --
13 1975 --          FUNCTION TWindow.FocusOnSuperView: BOOLEAN; OVERRIDE;
13 1976 --
13 1977 --          { Drawing Methods }
13 1978 --
13 1979 --          PROCEDURE TWindow.DrawContents; OVERRIDE;

```

```

13 1980 --
13 1981 --      PROCEDURE TWindow.Draw (area: Rect); OVERRIDE;
13 1982 --
13 1983 --      PROCEDURE TWindow.DrawResizeIcon;
13 1984 --
13 1985 --      { Mouse Handling Methods }
13 1986 --
13 1987 --      FUNCTION TWindow.HandleMouseDown (theMouse:      VPoint;
13 1988 --                                     VAR info:      EventInfo;
13 1989 --                                     VAR hysteresis: Point;
13 1990 --                                     VAR theCommand: TCommand); BOOLEAN; OVERRIDE;
13 1991 --
13 1992 --      { Menu Handling Methods }
13 1993 --
13 1994 --      FUNCTION TWindow.AllowsMenuAccess: BOOLEAN;
13 1995 --
13 1996 --      PROCEDURE TWindow.DoSetupMenus; OVERRIDE;
13 1997 --
13 1998 --      { Misc. Methods }
13 1999 --
13 2000 --      PROCEDURE TWindow.Show (state, redraw: BOOLEAN); OVERRIDE;
13 2001 --
13 2002 --      FUNCTION TWindow.IsShown: BOOLEAN; OVERRIDE;
13 2003 --
13 2004 --      PROCEDURE TWindow.Select;
13 2005 --
13 2006 --      PROCEDURE TWindow.SetTitleForDoc (newDocTitle: Str255);
13 2007 --
13 2008 --      PROCEDURE TWindow.AdaptToScreen;
13 2009 --
13 2010 --      PROCEDURE TWindow.ForceOnScreen;
13 2011 --
13 2012 --      PROCEDURE TWindow.Center (horizontally, vertically: BOOLEAN);
13 2013 --
13 2014 --      PROCEDURE TWindow.SimpleStagger (dh, dv: INTEGER; VAR counter: INTEGER);
13 2015 --
13 2016 --      FUNCTION TWindow.GetGrafPort: GrafPtr; OVERRIDE;
13 2017 --
13 2018 --      PROCEDURE TWindow.GetGlobalBounds (VAR globalBounds: Rect);
13 2019 --
13 2020 --      FUNCTION TWindow.GetWindow: TWindow; OVERRIDE;
13 2021 --
13 2022 --      PROCEDURE TWindow.SetResizeLimits (minSize, maxSize: Point);
13 2023 --
13 2024 --      PROCEDURE TWindow.SetTarget (newTarget: TEvtHandler);
13 2025 --
13 2026 --      PROCEDURE TWindow.InstallDocument (itsDocument: TDocument);
13 2027 --
13 2028 --      { Debugging Methods }
13 2029 --
13 2030 --      {$IFC qDebug}
13 2031 --      PROCEDURE TWindow.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2032 --                                     fieldAddr: Ptr;
13 2033 --                                     fieldType: INTEGER)); OVERRIDE;
13 2034 --
13 2035 --      PROCEDURE TWindow.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
13 2036 --      {$ENDC}
13 2037 --
13 2038 --      END { TWindow };
13 2039 --
13 2040 --      +-----+
13 2041 --      | TControl |
13 2042 --      +-----+
13 2043 --      ControlTemplatePtr = ^ControlTemplate;
13 2044 --      ControlTemplate = PACKED RECORD
13 2045 --          itsAdornment: Integer256;      { Cast to CntlAdornment }
13 2046 --          filler1:      0..255;
13 2047 --          itsPenSize:   Point;
13 2048 --          isSizable:   BOOLEAN;
13 2049 --          isDimmed:    BOOLEAN;
13 2050 --          isHilited:   BOOLEAN;
13 2051 --          canDismiss:  BOOLEAN;
13 2052 --          filler2:      0..4095;
13 2053 --          itsInset:    Rect;
13 2054 --          itsTextFace: Style;
13 2055 --          itsTextSize: INTEGER;
13 2056 --          itsTextColor: RGBColor;
13 2057 --          itsFontName: Str255;      {Actually a variable length P-String}
13 2058 --      END;
13 2059 --
13 2060 --      TControl =      OBJECT (TView)
13 2061 --
13 2062 --          fDefChoice:   INTEGER;
13 2063 --          fHilite:     BOOLEAN;
13 2064 --          fDimmed:     BOOLEAN;
13 2065 --          fSizeable:   BOOLEAN;
13 2066 --          fDismissesDialog: BOOLEAN;
13 2067 --          fAdornment:  CntlAdornment;
13 2068 --          fPenSize:    Point;
13 2069 --          fInset:      Rect;
13 2070 --          fTextStyle:  TextStyle;
13 2071 --
13 2072 --      {$IFC qProceduralViews}
13 2073 --      PROCEDURE TControl.TControl (itsSuperView: TView; itsLocation, itsSize: VPoint;
13 2074 --                                   itsHSizeDet, itsVSizeDet: SizeDeterminer);
13 2075 --      {$ENDC}
13 2076 --
13 2077 --      {$IFC qTemplateViews}
13 2078 --      PROCEDURE TControl.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;

```

```

13 2079 --      {$ENDC}
13 2080 --
13 2081 --      {$IFC qWriteTemplates}
13 2082 --          PROCEDURE TControl.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2083 --
13 2084 --          PROCEDURE TControl.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2085 --      {$ENDC}
13 2086 --
13 2087 --          PROCEDURE TControl.Activate (entering: BOOLEAN); OVERRIDE;
13 2088 --
13 2089 --          PROCEDURE TControl.ComputeSize (VAR newSize: VPoint); OVERRIDE;
13 2090 --
13 2091 --          FUNCTION TControl.ContainsMouse (theMouse: VPoint): BOOLEAN; OVERRIDE;
13 2092 --
13 2093 --          PROCEDURE TControl.ControlArea (VAR theArea: Rect);
13 2094 --
13 2095 --          PROCEDURE TControl.Dim;
13 2096 --
13 2097 --          PROCEDURE TControl.DimState (state, redraw: BOOLEAN);
13 2098 --
13 2099 --          FUNCTION TControl.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
13 2100 --              VAR hysteresis: Point): TCommand; OVERRIDE;
13 2101 --
13 2102 --          PROCEDURE TControl.Draw (area: Rect); OVERRIDE;
13 2103 --
13 2104 --          FUNCTION TControl.Focus: BOOLEAN; OVERRIDE;
13 2105 --
13 2106 --          PROCEDURE TControl.Hilite;
13 2107 --
13 2108 --          PROCEDURE TControl.HiliteState (state, redraw: BOOLEAN);
13 2109 --
13 2110 --          PROCEDURE TControl.Inset (dh, dv: INTEGER; redraw: BOOLEAN);
13 2111 --
13 2112 --          FUNCTION TControl.IsDimmed: BOOLEAN;
13 2113 --
13 2114 --          PROCEDURE TControl.SetInset (newInset: Rect; redraw: BOOLEAN);
13 2115 --
13 2116 --          PROCEDURE TControl.InstallColor (theColor: RGBColor; redraw: BOOLEAN);
13 2117 --
13 2118 --          PROCEDURE TControl.InstallTextStyle (theTextStyle: TextStyle; redraw: BOOLEAN);
13 2119 --
13 2120 --          PROCEDURE TControl.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 2121 --
13 2122 --          PROCEDURE TControl.TrackConstrain(anchorPoint, previousPoint: VPoint;
13 2123 --              VAR nextPoint: VPoint);
13 2124 --              { Override this if you want to constrain the mouse point to a grid,
13 2125 --                force drawing a square, etc. This is called only if
13 2126 --                fConstrainsMouse is TRUE. }
13 2127 --
13 2128 --          PROCEDURE TControl.TrackFeedback(anchorPoint, nextPoint: VPoint;
13 2129 --              turnItOn, mouseDidMove: BOOLEAN);
13 2130 --
13 2131 --          PROCEDURE TControl.TrackMouse(aTrackPhase: TrackPhase;
13 2132 --              VAR anchorPoint, previousPoint, nextPoint: VPoint;
13 2133 --              mouseDidMove: BOOLEAN);
13 2134 --
13 2135 --          FUNCTION TControl.Validate: BOOLEAN;
13 2136 --
13 2137 --      {$IFC qDebug}
13 2138 --          PROCEDURE TControl.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2139 --              fieldAddr: Ptr;
13 2140 --              fieldType: INTEGER)); OVERRIDE;
13 2141 --      {$ENDC}
13 2142 --
13 2143 --          END { TControl };
13 2144 --
13 2145 --      {-----+
13 2146 --      | TctlMgr |
13 2147 --      +-----+}
13 2148 --      TctlMgr = OBJECT (TControl)
13 2149 --
13 2150 --          fCmgrControl:      ControlHandle;
13 2151 --
13 2152 --      {$IFC qProceduralViews}
13 2153 --          PROCEDURE TctlMgr.ICtlMgr (itsSuperView: TView; itsLocation, itsSize: VPoint;
13 2154 --              itsHSizeDet, itsVSizeDet: SizeDeterminer;
13 2155 --              itsTitle: Str255; itsVal, itsMin, itsMax, itsProcID: INTEGER);
13 2156 --      {$ENDC}
13 2157 --
13 2158 --      {$IFC qTemplateViews}
13 2159 --          PROCEDURE TctlMgr.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
13 2160 --
13 2161 --          PROCEDURE TctlMgr.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2162 --
13 2163 --          PROCEDURE TctlMgr.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2164 --      {$ENDC}
13 2165 --
13 2166 --          PROCEDURE TctlMgr.Free; OVERRIDE;
13 2167 --
13 2168 --          PROCEDURE TctlMgr.BeInPort (itsPort: GrafPtr); OVERRIDE;
13 2169 --
13 2170 --          PROCEDURE TctlMgr.CreateCmgrControl (itsBounds: Rect; itsTitle: Str255;
13 2171 --              itsValue, itsMin, itsMax, itsProcID: INTEGER);
13 2172 --
13 2173 --          PROCEDURE TctlMgr.DimState (state, redraw: BOOLEAN); OVERRIDE;
13 2174 --
13 2175 --          FUNCTION TctlMgr.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
13 2176 --              VAR hysteresis: Point): TCommand; OVERRIDE;
13 2177 --

```

```

13 2178 --      PROCEDURE TCtrlMgr.Draw (area: Rect); OVERRIDE;
13 2179 --
13 2180 --      FUNCTION TCtrlMgr.GetMax: INTEGER;
13 2181 --
13 2182 --      FUNCTION TCtrlMgr.GetMin: INTEGER;
13 2183 --
13 2184 --      PROCEDURE TCtrlMgr.GetText (VAR theText: Str255);
13 2185 --
13 2186 --      FUNCTION TCtrlMgr.GetVal: INTEGER;
13 2187 --
13 2188 --      PROCEDURE TCtrlMgr.HiliteState (state, redraw: BOOLEAN); OVERRIDE;
13 2189 --
13 2190 --      PROCEDURE TCtrlMgr.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
13 2191 --
13 2192 --      PROCEDURE TCtrlMgr.SetMax (itsMax: INTEGER; redraw: BOOLEAN);
13 2193 --
13 2194 --      PROCEDURE TCtrlMgr.SetMin (itsMin: INTEGER; redraw: BOOLEAN);
13 2195 --
13 2196 --      PROCEDURE TCtrlMgr.SetText (itsText: Str255; redraw: BOOLEAN);
13 2197 --
13 2198 --      PROCEDURE TCtrlMgr.SetVal (newVal: INTEGER; redraw: BOOLEAN);
13 2199 --
13 2200 --      PROCEDURE TCtrlMgr.SetValues (itsVal, itsMin, itsMax: INTEGER; redraw: BOOLEAN);
13 2201 --
13 2202 --      PROCEDURE TCtrlMgr.WhileFocused (PROCEDURE DoToControl; redraw: BOOLEAN);
13 2203 --
13 2204 --      { $IFC qDebug }
13 2205 --      PROCEDURE TCtrlMgr.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2206 --      fieldAddr: Ptr;
13 2207 --      fieldType: INTEGER)); OVERRIDE;
13 2208 --
13 2209 --      { $ENDC }
13 2210 --      END { TCtrlMgr };
13 2211 --
13 2212 --      {-----+
13 2213 --      | TScrollBar |
13 2214 --      +-----+
13 2215 --      ScrollBarTemplatePtr = ^ScrollBarTemplate;
13 2216 --      ScrollBarTemplate =
13 2217 --      RECORD
13 2218 --      value:      LONGINT;
13 2219 --      minimum:    LONGINT;
13 2220 --      maximum:    LONGINT;
13 2221 --      END;
13 2222 --
13 2223 --      TScrollBar = OBJECT (TCtrlMgr)
13 2224 --
13 2225 --      fBitsToShift:    INTEGER;    { # bits shifted to convert between
13 2226 --      short and long value. }
13 2227 --
13 2228 --      fDirection:      VHSelect;
13 2229 --      fLongVal:         VCoordinate;
13 2230 --      fLongMin:         VCoordinate;
13 2231 --      fLongMax:         VCoordinate;
13 2232 --
13 2233 --      { $IFC qProceduralViews }
13 2234 --      PROCEDURE TScrollBar.IScrollBar (itsSuperview: TView; itsLocation, itsSize: VPoint;
13 2235 --      itsHSizeDet, itsVSizeDet: SizeDeterminer;
13 2236 --      itsDirection: VHSelect; itsVal, itsMin, itsMax: LONGINT);
13 2237 --
13 2238 --      { $ENDC }
13 2239 --
13 2240 --      { $IFC qTemplateViews }
13 2241 --      PROCEDURE TScrollBar.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
13 2242 --
13 2243 --      { $ENDC }
13 2244 --
13 2245 --      { $IFC qWriteTemplates }
13 2246 --      PROCEDURE TScrollBar.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2247 --
13 2248 --      { $ENDC }
13 2249 --
13 2250 --      PROCEDURE TScrollBar.DeltaValue (delta: VCoordinate);
13 2251 --
13 2252 --      FUNCTION TScrollBar.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
13 2253 --      VAR hysteresis: Point): TCommand; OVERRIDE;
13 2254 --
13 2255 --      FUNCTION TScrollBar.GetLongMax: VCoordinate;
13 2256 --
13 2257 --      FUNCTION TScrollBar.GetLongMin: VCoordinate;
13 2258 --
13 2259 --      FUNCTION TScrollBar.GetLongVal: VCoordinate;
13 2260 --
13 2261 --      PROCEDURE TScrollBar.SetLongMax (itsMax: VCoordinate; redraw: BOOLEAN);
13 2262 --
13 2263 --      PROCEDURE TScrollBar.SetLongMin (itsMin: VCoordinate; redraw: BOOLEAN);
13 2264 --
13 2265 --      PROCEDURE TScrollBar.SetLongVal (itsVal: VCoordinate; redraw: BOOLEAN);
13 2266 --
13 2267 --      PROCEDURE TScrollBar.SetLongValues (itsVal, itsMin, itsMax: VCoordinate; redraw: BOOLEAN);
13 2268 --
13 2269 --      PROCEDURE TScrollBar.TrackScrollBar (partCode: INTEGER);
13 2270 --
13 2271 --      { Debugging Methods }
13 2272 --
13 2273 --      { $IFC qDebug }
13 2274 --      PROCEDURE TScrollBar.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2275 --      fieldAddr: Ptr;
13 2276 --      fieldType: INTEGER)); OVERRIDE;
13 2277 --
13 2278 --      { $ENDC }
13 2279 --      END { TScrollBar };
13 2280 --
13 2281 --      {-----+

```

```

13 2277 --      | TSScrollBar                                |
13 2278 --      +-----+
13 2279 --      TSScrollBar = OBJECT (TScrollBar)
13 2280 --
13 2281 --          fScrollers:          TList;
13 2282 --
13 2283 --      ($IFC qProceduralViews)
13 2284 --          PROCEDURE TSScrollBar.ISScrollBar (itsSuperview: TView; itsLocation, itsSize: VPoint;
13 2285 --              itsHSizeDet, itsVSizeDet: SizeDeterminer;
13 2286 --              itsDirection: VHSelect; itsMax: LONGINT;
13 2287 --              itsScroller: TScroller);
13 2288 --      ($ENDC)
13 2289 --
13 2290 --      ($IFC qTemplateViews)
13 2291 --          PROCEDURE TSScrollBar.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
13 2292 --      ($ENDC)
13 2293 --
13 2294 --      ($IFC qWriteTemplates)
13 2295 --          PROCEDURE TSScrollBar.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2296 --
13 2297 --          PROCEDURE TSScrollBar.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
13 2298 --      ($ENDC)
13 2299 --
13 2300 --          PROCEDURE TSScrollBar.Free; OVERRIDE;
13 2301 --
13 2302 --          PROCEDURE TSScrollBar.Activate (entering: BOOLEAN); OVERRIDE;
13 2303 --
13 2304 --          FUNCTION TSScrollBar.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
13 2305 --              VAR hysteresis: Point): TCommand; OVERRIDE;
13 2306 --
13 2307 --          PROCEDURE TSScrollBar.Draw (area: Rect); OVERRIDE;
13 2308 --
13 2309 --          PROCEDURE TSScrollBar.TrackScrollBar (partCode: INTEGER); OVERRIDE;
13 2310 --
13 2311 --      { Debugging Methods }
13 2312 --
13 2313 --      ($IFC qDebug)
13 2314 --          PROCEDURE TSScrollBar.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2315 --              fieldAddr: Ptr;
13 2316 --              fieldType: INTEGER)); OVERRIDE;
13 2317 --      ($ENDC)
13 2318 --          END { TSScrollBar };
13 2319 --
13 2320 --
13 2321 --      +-----+
13 2322 --      | TPrintHandler                                |
13 2323 --      +-----+
13 2324 --      TPrintHandler = OBJECT (TEvtHandler)
13 2325 --
13 2326 --          fView:          TView;          { The view whose printing is handled }
13 2327 --          fDeviceRes:      Point;          { formal printer resolution, spots
13 2328 --              per inch }
13 2329 --          fEffectiveDeviceRes: Point;      { true effective spots per inch of
13 2330 --              printer; larger than the fDeviceRes if
13 2331 --              REDUCTION is in effect, smaller if
13 2332 --              ENLARGEMENT is, same as fDeviceRes if
13 2333 --              neither is in effect. }
13 2334 --          fViewPerPage:    VPoint;        { The amount of the view that is visible on a page }
13 2335 --          fFocusedPage:    INTEGER;       { The page number currently focused }
13 2336 --
13 2337 --      { Initialization and termination }
13 2338 --
13 2339 --          PROCEDURE TPrintHandler.IPrintHandler (itsView: TView);
13 2340 --              { Initialize the printHandler, and associate it and 'itsView' with
13 2341 --              each other }
13 2342 --
13 2343 --      ($IFC qDebug)
13 2344 --          PROCEDURE TPrintHandler.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
13 2345 --
13 2346 --          PROCEDURE TPrintHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2347 --              fieldAddr: Ptr;
13 2348 --              fieldType: INTEGER)); OVERRIDE;
13 2349 --      ($ENDC)
13 2350 --
13 2351 --          PROCEDURE TPrintHandler.FocusOnInterior;
13 2352 --
13 2353 --      { Printing-related reformatting and computations }
13 2354 --
13 2355 --          FUNCTION TPrintHandler.BreakFollowing (vhs: VHSelect;
13 2356 --              prevBreak: VCoordinate; VAR automatic: BOOLEAN): VCoordinate;
13 2357 --              { Returns the location of the page break which follows the page break
13 2358 --              located at 'prevBreak', in direction vhs; returns automatic =
13 2359 --              TRUE if the page-break is thought of as an 'automatic' rather
13 2360 --              than a 'manual' (user-specified) one. Note that page-breaks in
13 2361 --              dimension 'v' are drawn as vertical lines, those in dimension
13 2362 --              'h' as horizontal lines }
13 2363 --
13 2364 --          PROCEDURE TPrintHandler.CalcPageStrips (VAR pageStrips: Point);
13 2365 --              { Recalculate the number of strips of pages in each dimension }
13 2366 --
13 2367 --          PROCEDURE TPrintHandler.CalcViewPerPage (VAR amtPerPage: VPoint);
13 2368 --              { Computes the amount of view, in each dimension, to be allocated to
13 2369 --              a printed page }
13 2370 --
13 2371 --          PROCEDURE TPrintHandler.CheckPrinter;
13 2372 --              { See if there are changed printer-configuration parameters which
13 2373 --              need to be absorbed, and if so, absorb them }
13 2374 --
13 2375 --          PROCEDURE TPrintHandler.LocatePageInterior (pageNumber: INTEGER;

```

```

13 2376 --      VAR loc: Point);
13 2377 --      { Decide where the top-left-most point of the the interior of the
13 2378 --      page should be }
13 2379 --
13 2380 --      PROCEDURE TPrintHandler.PrinterChanged;
13 2381 --      { The metrics relating to printer use have changed; absorb the
13 2382 --      information and react }
13 2383 --
13 2384 --      PROCEDURE TPrintHandler.RedoPageBreaks;
13 2385 --      { Recompute the dividing lines between areas of the view which will
13 2386 --      be mapped into different printed pages }
13 2387 --
13 2388 --      PROCEDURE TPrintHandler.Reset;
13 2389 --      { Resets the print handler to the default values }
13 2390 --
13 2391 --      { Printing-related commands }
13 2392 --
13 2393 --      FUNCTION TPrintHandler.Print(itsCmdNumber: CmdNumber; VAR proceed: BOOLEAN): TCommand; (+)
13 2394 --
13 2395 --      FUNCTION TPrintHandler.SetupForFinder: BOOLEAN;
13 2396 --      { Intended to set up the print handler for finder printing. }
13 2397 --
13 2398 --
13 2399 --      { Printing-related screen feedback }
13 2400 --
13 2401 --      PROCEDURE TPrintHandler.DrawPrintFeedback(area: Rect);
13 2402 --      { Draw page-breaks, page-numbers, view-borders, rulers, etc. }
13 2403 --
13 2404 --      PROCEDURE TPrintHandler.DrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
13 2405 --      loc: VCoordinate; automatic: BOOLEAN);
13 2406 --
13 2407 --      { Actual printing }
13 2408 --
13 2409 --      FUNCTION TPrintHandler.MaxPageNumber: INTEGER;
13 2410 --      { Returns the largest page number which could be reasonably printed,
13 2411 --      given the properties of the view; in cases like a SpreadSheet,
13 2412 --      this is not necessarily as large as the product (row strips) x
13 2413 --      (column strips) }
13 2414 --
13 2415 --      PROCEDURE TPrintHandler.SetPageInterior(pageNumber: INTEGER);
13 2416 --      { Set up the pad-space interior rectangle for the given page number }
13 2417 --
13 2418 --      PROCEDURE TPrintHandler.SetPageOffset(coord: VPoint);
13 2419 --      { Given the view coordinates of the top-left-most point of the view
13 2420 --      which is being mapped into the current page (in coord), this
13 2421 --      method's job is to compute the correct fRelOrigin field for the
13 2422 --      current page }
13 2423 --
13 2424 --      END;
13 2425 --
13 2426 --
13 2427 --      {-----+
13 2428 --      | TCommand |
13 2429 --      +-----+
13 2430 --      TCommand = OBJECT(TObject)
13 2431 --
13 2432 --      fCmdNumber:      CmdNumber;      { Can be a many-to-one mapping of
13 2433 --      fCmdNumber to TCommand }
13 2434 --      fChangedDocument: TDocument;      { the document changed by this
13 2435 --      command; defaults to gDocument
13 2436 --      global variable }
13 2437 --      fTarget:          TEvtHandler;    { The target tp set before calling
13 2438 --      UndoIt or RedoIt }
13 2439 --      fCmdDone:         BOOLEAN;        { TRUE if the last "execution"
13 2440 --      message sent to the command was
13 2441 --      DoIt or RedoIt }
13 2442 --      fCanUndo:         BOOLEAN;        { Defaults to TRUE }
13 2443 --      fCausesChange:    BOOLEAN;        { Defaults to TRUE; Marks document
13 2444 --      changed when command is done }
13 2445 --      fChangesClipboard: BOOLEAN;        { Defaults to FALSE. Set to true for
13 2446 --      command subclasses representing
13 2447 --      CUT and/ or COPY commands which
13 2448 --      change the Clipboard. }
13 2449 --      { The remaining fields are for command objects that track the mouse }
13 2450 --      fView:            TView;          { The view in which tracking takes place }
13 2451 --      fConstrainsMouse: BOOLEAN;        { Defaults to FALSE; if you set to
13 2452 --      TRUE then TrackConstrain will be
13 2453 --      called to constrain the mouse. }
13 2454 --      fViewConstrain:   BOOLEAN;        { Defaults to TRUE, which means
13 2455 --      constrain mouse to view limits }
13 2456 --      fTrackNonMovement: BOOLEAN;        { Defaults to FALSE, which means don't
13 2457 --      track mouse unless it moves. }
13 2458 --      fScroller:        TScroller;      { The scroller that handles auto-scrolling }
13 2459 --
13 2460 --      { Init & Free }
13 2461 --      PROCEDURE TCommand.ICommand(itsCmdNumber: CmdNumber; itsDocument: TDocument;
13 2462 --      itsView: TView; itsScroller: TScroller);
13 2463 --
13 2464 --      ($IFC qDebug)
13 2465 --      { Debugging }
13 2466 --      PROCEDURE TCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2467 --      fieldAddr: Ptr;
13 2468 --      fieldType: INTEGER)); OVERRIDE;
13 2469 --
13 2470 --      {SENDC}
13 2471 --
13 2472 --      { command processing - usually overridden }
13 2473 --
13 2474 --      PROCEDURE TCommand.Commit;
13 2475 --      PROCEDURE TCommand.DoIt;

```

```

13 2475 --      PROCEDURE TCommand.RedoIt;
13 2476 --      PROCEDURE TCommand.UndoIt;
13 2477 --
13 2478 --      ( Mouse Tracking )
13 2479 --
13 2480 --      PROCEDURE TCommand.TrackConstrain(anchorPoint, previousPoint: VPoint;
13 2481 --          VAR nextPoint: VPoint);
13 2482 --          ( Override this if you want to constrain the mouse point to a grid,
13 2483 --            force drawing a square, etc. This is called only if
13 2484 --            fConstrainsMouse is TRUE. )
13 2485 --
13 2486 --      PROCEDURE TCommand.TrackFeedback(anchorPoint, nextPoint: VPoint;
13 2487 --          turnItOn, mouseDidMove: BOOLEAN);
13 2488 --          ( Default is: IF NOT mouseDidMove THEN FrameRect(<rect formed by
13 2489 --            anchorPoint and nextPoint>). Before this is called the pen
13 2490 --            is set to PenNormal and the PenMode to PatXOR. )
13 2491 --
13 2492 --      FUNCTION TCommand.TrackMouse(aTrackPhase: TrackPhase;
13 2493 --          VAR anchorPoint, previousPoint, nextPoint: VPoint;
13 2494 --          mouseDidMove: BOOLEAN): TCommand;
13 2495 --          ( This does the mouse tracking.
13 2496 --
13 2497 --      The default returns SELF. Override it to force gridding, to return
13 2498 --      a different type of command, or to perform the command on mouse
13 2499 --      release instead of using the usual DoIt approach.
13 2500 --
13 2501 --      All points are in local coordinates.
13 2502 --
13 2503 --      If aTrackPhase = trackPress then all 3 points will be the same.
13 2504 --      If aTrackPhase = trackRelease then nextPoint will be the coordinate
13 2505 --      in the mouseUp event (if a mouseUp event is found), otherwise
13 2506 --      the same point as previousPoint. Generally, you will ignore
13 2507 --      nextPoint and just look at previousPoint.
13 2508 --
13 2509 --      mouseDidMove will be TRUE if aTrackPhase = trackPress or
13 2510 --      trackRelease; otherwise, it will indicate if nextPoint =
13 2511 --      previousPoint.
13 2512 --
13 2513 --      If you change anchorPoint, the new value will be passed to you next
13 2514 --      time. The value of nextPoint at the time the routine exits will
13 2515 --      be passed to you as previousPoint the next time the routine is
13 2516 --      called. You can change nextPoint if you wish.
13 2517 --      Usually, however, you will do gridding in the TrackConstrain method. )
13 2518 --
13 2519 --      PROCEDURE TCommand.AutoScroll (deltaH, deltaV: VCoordinate);
13 2520 --          ( Implements autoscrolling by calling fScroller.ScrollBy. )
13 2521 --
13 2522 --      END;
13 2523 --
13 2524 --
13 2525 --      {-----+
13 2526 --      | TControlTracker |
13 2527 --      +-----+
13 2528 --      TControlTracker = OBJECT (TCommand)
13 2529 --
13 2530 --          fControl: TControl;
13 2531 --
13 2532 --      PROCEDURE TControlTracker.IControlTracker (theControl: TControl);
13 2533 --
13 2534 --      FUNCTION TControlTracker.TrackMouse(aTrackPhase: TrackPhase;
13 2535 --          VAR anchorPoint, previousPoint, nextPoint: VPoint;
13 2536 --          mouseDidMove: BOOLEAN): TCommand; OVERRIDE;
13 2537 --
13 2538 --      PROCEDURE TControlTracker.TrackFeedback(anchorPoint, nextPoint: VPoint;
13 2539 --          turnItOn, mouseDidMove: BOOLEAN); OVERRIDE;
13 2540 --
13 2541 --      PROCEDURE TControlTracker.TrackConstrain (anchorPoint, previousPoint: VPoint;
13 2542 --          VAR nextPoint: VPoint); OVERRIDE;
13 2543 --
13 2544 --      END { TControlTracker };
13 2545 --
13 2546 --
13 2547 --      {-----+
13 2548 --      | TDeskScrapView |
13 2549 --      +-----+
13 2550 --      TDeskScrapView = OBJECT(TView)
13 2551 --
13 2552 --          fHavePicture: BOOLEAN;
13 2553 --          fHaveText: BOOLEAN;
13 2554 --          fScrapCount: INTEGER; ( scrapCount of the scrap represented by
13 2555 --                                this view.)
13 2556 --
13 2557 --          fDataHandle: Handle; ( if non-NIL, will be either a PicHandle to
13 2558 --                                my picture or a Handle to my Text )
13 2559 --
13 2560 --      ($IFC qProceduralViews)
13 2561 --          PROCEDURE TDeskScrapView.IDeskScrapView;
13 2562 --      ($ENDC)
13 2563 --
13 2564 --      ($IFC qTemplateViews)
13 2565 --          PROCEDURE TDeskScrapView.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
13 2566 --      ($ENDC)
13 2567 --
13 2568 --      PROCEDURE TDeskScrapView.Free; OVERRIDE;
13 2569 --
13 2570 --      PROCEDURE TDeskScrapView.CheckScrapContents;
13 2571 --      PROCEDURE TDeskScrapView.CalcMinSize (VAR minSize: VPoint); OVERRIDE;
13 2572 --      PROCEDURE TDeskScrapView.Draw(area: Rect); OVERRIDE;
13 2573 --      PROCEDURE TDeskScrapView.SuperViewChangedSize (delta: VPoint; invalidate: BOOLEAN); OVERRIDE;

```

```

13 2574 --      PROCEDURE TDeskScrapView.WriteToDeskScrap; OVERRIDE;
13 2575 --
13 2576 --      ($IFC qDebug)
13 2577 --      ( Debugging )
13 2578 --      PROCEDURE TDeskScrapView.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
13 2579 --
13 2580 --      PROCEDURE TDeskScrapView.Fields (PROCEDURE DoToField (fieldName: Str255;
13 2581 --      fieldAddr: Ptr;
13 2582 --      fieldType: INTEGER)); OVERRIDE;
13 2583 --
13 2584 --      ($SENDC)
13 2585 --
13 2586 --      END;
13 2587 --
13 2588 --      {-----+
13 2589 --      | Global Variables |
13 2590 --      +-----}
13 2591 --      VAR
13 2592 --      gAppDone:          BOOLEAN;          { set this to TRUE when you want the
13 2593 --      application to terminate }
13 2594 --      gApplication:      TApplication;      { the application object }
13 2595 --      ($IFC qDebug)
13 2596 --      gBusyTempRgn:      BOOLEAN;          { Is gTempRgn in use? }
13 2597 --      ($SENDC)
13 2598 --      gChooserOK:        BOOLEAN;          { if FALSE, user will not be allowed
13 2599 --      to change the printer selection using
13 2600 --      'Choose Printer' DA while the application is
13 2601 --      alive; if TRUE (the default set in
13 2602 --      InitToolBox), then printer-change will only
13 2603 --      be disabled during background printing. If
13 2604 --      you want it FALSE in your application, the
13 2605 --      right time to set it is after calling
13 2606 --      InitToolBox (where it is initialized to
13 2607 --      TRUE) and before calling InitPrinting
13 2608 --      (where, if it's found to be FALSE, the low-
13 2609 --      memory location governing is option is poked) }
13 2610 --      gClickCount:       INTEGER;          { number of 'saved up' mouse clicks; handled in
13 2611 --      TApplication.DispatchEvent, ObeyEvent &
13 2612 --      ObeyMouseDown.
13 2613 --      If a mouseDown, the value is that returned by
13 2614 --      the call to TApplication.CountClicks.
13 2615 --      If a mouseUp, the value is left alone.
13 2616 --      Otherwise, set to 0.
13 2617 --      Also set to 0 is the 2 clicks were in different
13 2618 --      parts of the window. }
13 2619 --      gClipClaimed:      BOOLEAN;          { Used by PerformCommand & ClaimClipboard to determine, if
13 2620 --      DoIt of a cut/copy cmd fails, whether the Clipboard had
13 2621 --      already been claimed by the new command or not }
13 2622 --      gClipOrphanage:    TDeskScrapView;   { A view to represent the Clipboard
13 2623 --      when it contains TEXT or PICT
13 2624 --      data }
13 2625 --
13 2626 --      gClipUndoView:     TView;            { The view previously installed in
13 2627 --      the Clipboard }
13 2628 --
13 2629 --      gClipView:         TView;            { The view currently installed in the
13 2630 --      Clipboard }
13 2631 --      gClipWindow:       TWindow;          { The window holding the Clipboard
13 2632 --      display }
13 2633 --      gClipWrittenToDeskScrap: BOOLEAN;     { True if the clipboard view has been
13 2634 --      written to the desk scrap. }
13 2635 --      gCouldPrint:       BOOLEAN;          { whether Printer code is accessible
13 2636 --      to the application }
13 2637 --      gCurrPrintHandler: TPrintHandler;    { If printing, this is set to the print
13 2638 --      handler. Nil if not printing. }
13 2639 --      gCursorRgn:        RgnHandle;        { the current cursor region }
13 2640 --      ($IFC qDebug)
13 2641 --      gDebugPrinting:    BOOLEAN;          { simple toggle for debugging printing }
13 2642 --      ($SENDC)
13 2643 --      gDocument:         TDocument;        { the current document }
13 2644 --      gDocList:          TList;            { list of documents }
13 2645 --      gDrawingPictScrap: BOOLEAN;          { TRUE if a view's Draw routine is
13 2646 --      being called in order to create PICT data in
13 2647 --      the Desk Scrap; your View.Draw can check
13 2648 --      this routine, and insert PictComments as
13 2649 --      appropriate if it wishes to have them in
13 2650 --      the Picture in the Desk Scrap }
13 2651 --      gErrorParm3:       Str255;          { This is used as the last argument
13 2652 --      to ParamText in ErrorAlert, if ErrorAlert
13 2653 --      displays one of the standard alerts. (This
13 2654 --      string will replace ^3 in those alerts.)
13 2655 --      ErrorAlert also sets this to '' when called.
13 2656 --      You can use this to parameterize the
13 2657 --      automatic alerts that MacApp displays. For
13 2658 --      example, TDocument.ReadFromFile sets this to
13 2659 --      the document name. }
13 2660 --      gEventLevel:       INTEGER;          { A count of the number of nested calls to
13 2661 --      PollEvent. }
13 2662 --      ($IFC qDebug)
13 2663 --      gExperimenting:    BOOLEAN;          { simple toggle for enabling/disabling
13 2664 --      experimental features }
13 2665 --      ($SENDC)
13 2666 --      gFakeWindow:       WindowRecord;     { Initialized with OpenPort, and by
13 2667 --      setting its Controllist to NIL.
13 2668 --      This is NOT a real window, just
13 2669 --      a port. }
13 2670 --      gFileCount:        INTEGER;          { # files to open/print from finder; set in Init2 }
13 2671 --      gFinderPrinting:   BOOLEAN;          { TRUE if the Finder started the
13 2672 --      App up just for printing docs }

```



```

13 2673 --      gFocusedView:      TView;      { the view that is currently focused }
13 2674 --      gFreeWindowList:    TList;      { list of free-standing
13 2675 --                               (documentless) windows }
13 2676 --      gFrontWindow:      TWindow;      { If non-NIL the TWindow object of
13 2677 --                               the FrontWindow. }
13 2678 --      gGotClipType:      BOOLEAN;      { True if we can paste the data in the clipboard }
13 2679 --      gHeadCohandler:    TEvtHandler;  { head of linked list of global
13 2680 --                               co-handlers }
13 2681 --      gIdleFreq:          LONGINT;      { number of ticks needed to call DoIdle's
13 2682 --                               since the last DoIdle's. }
13 2683 --      gIdlePhase:        IdlePhase;     { The current idle phase }
13 2684 --      gInBackground:      BOOLEAN;      { True if app is currently in background }
13 2685 --      gInitialized:      BOOLEAN;      { Set to TRUE at the end of
13 2686 --                               IApplication }
13 2687 --      { $IFC qDebug }
13 2688 --      gIntenseDebugging:  BOOLEAN;      { debugging toggle for intensive debugging }
13 2689 --      { $ENDC }
13 2690 --      gLastClickPart:    INTEGER;      { the list window part clicked in-used for double
13 2691 --                               click detection }
13 2692 --      gLastCommand:      TCommand;      { the last command done or undone by
13 2693 --                               the user }
13 2694 --      gLastDeskAcc:      LONGINT;      { time of the most recent possible
13 2695 --                               excursion to a Desk Accessory }
13 2696 --      gLastIdle:         LONGINT;      { time in ticks of the last time the
13 2697 --                               event handlers were idled. }
13 2698 --      gLastMsePt:        Point;         { coordinates of mouse in last event
13 2699 --                               passed to TApplication.CountClicks }
13 2700 --      gLastUpTime:       LONGINT;      { time of last mouse up event passed
13 2701 --                               to TApplication.ObeyEvent }
13 2702 --      gLongOffset:       VPoint;
13 2703 --      gLowSpaceInterval: LONGINT;      { If >= 0, the frequency (in Ticks)
13 2704 --                               with which MacApp displays a low
13 2705 --                               space alert. (Defaults to
13 2706 --                               kLowSpaceInterval.) If < 0,
13 2707 --                               MacApp doesn't display an alert. }
13 2708 --      gMainEventMask:    INTEGER;      { Event mask used in main event loop.
13 2709 --                               Initialized by InitToolbox. }
13 2710 --      gMainFileType:     OSType;      { principal file type opened/printed
13 2711 --                               by application; set in
13 2712 --                               TApplication.IApplication; by default,
13 2713 --                               TApplication.SFGetFilters returns a list of
13 2714 --                               just this }
13 2715 --      gMBarDisplayed:    INTEGER;      { menus that are read in and installed in
13 2716 --                               menu bar }
13 2717 --      gMBarHeight:       INTEGER;      { Height of the menu bar in pixels }
13 2718 --      gMBarHierarchical: INTEGER;      { menus that pop up when a menu item is
13 2719 --                               choosen }
13 2720 --      gMBarNotDisplayed: INTEGER;      { menus that are read in but not installed }
13 2721 --      gNewScrapStuff:    ScrapStuff;
13 2722 --      gNextSpaceMsg:     LONGINT;      { time when next low space message should
13 2723 --                               be displayed }
13 2724 --      gNoChanges:        TCommand;      { special TCommand object to return
13 2725 --                               if the command does not change the document.
13 2726 --                               You should carry out the command in the
13 2727 --                               DoMenuCommand procedure. }
13 2728 --      gNullPrintHandler: TPrintHandler; { handles printing-relating messages
13 2729 --                               for views which don't print }
13 2730 --      gNumUntitled:      INTEGER;      { The number to assign to the next
13 2731 --                               Untitled document (assuming that the app
13 2732 --                               provides a template for filling in the
13 2733 --                               number. You must change STR# resource with
13 2734 --                               ID 0 / index 3 to read (for example)
13 2735 --                               Untitled-<##>>>. }
13 2736 --      gOldChooserFlag:   BOOLEAN;      { The state of the chooser alert flag when
13 2737 --                               the application started up. }
13 2738 --      gOldScrapStuff:    ScrapStuff;
13 2739 --      gOrthogonal:       ARRAY[VHSelect] OF VHSelect;
13 2740 --      gPageOffset:       VPoint;      { offset in view of page being
13 2741 --                               printed }
13 2742 --      gPrefClipType:     ResType;
13 2743 --      gPrintHandler:     TPrintHandler; { a global print-handler object for
13 2744 --                               use in some standard printing-related
13 2745 --                               activities: this is initialized to be just
13 2746 --                               a reference to gNullPrintHandler, but if
13 2747 --                               you call InitPrinting, that will install a
13 2748 --                               non-trivial print-handler here. }
13 2749 --      gPrinting:         BOOLEAN;      { true if currently Printing }
13 2750 --      { $IFC qTemplateViews }
13 2751 --      gProtoList:        TAssociation;  { List of object prototypes for cloning }
13 2752 --      { $ENDC }
13 2753 --      { $IFC qDebug }
13 2754 --      gReportEvt:        BOOLEAN;      { debugging toggle for reporting events }
13 2755 --      gReportMenuChoices: BOOLEAN;      { debugging toggle for tracing cmds }
13 2756 --      gRsrcCheck:        INTEGER;      { debugging toggle for checking resource
13 2757 --                               usage }
13 2758 --      { $ENDC }
13 2759 --
13 2760 --      { Set by GetFocus; used by SetFocus. GetFocus/SetFocus are used to speed up
13 2761 --      scrolling (see TScroller). }
13 2762 --      gSaveClip:         RgnHandle;
13 2763 --      gSaveFocusedView:  TView;
13 2764 --      gSaveOrg:          Point;
13 2765 --      gSaveLongOffset:   VPoint;
13 2766 --      gSavePort:         WindowPtr;
13 2767 --
13 2768 --      { $IFC qTemplateViews }
13 2769 --      gSignatures:       ARRAY [1..kMaxSignatures] OF IDType;      { Standard view signatures }
13 2770 --      gSignatureViews:   ARRAY [1..kMaxSignatures] OF TObj;
13 2771 --      gSignatureCount:    INTEGER;

```

```

13 2772 --      ($ENDC)
13 2773 --
13 2774 --      gStdHysteresis:    Point;      { standard hysteresis used in TCommand.ICommand }
13 2775 --      gStdStaggerCount:  INTEGER;    { Used to stagger windows created from templates }
13 2776 --      gStdWMoveBounds:   Rect;       { standard boundsRect to pass to DragWindow
13 2777 --                                   Toolbox routine }
13 2778 --      gStdWSizeRect:     Rect;       { standard sizeRect to pass to GrowWindow
13 2779 --                                   Toolbox routine }
13 2780 --      gStdWScreenRect:   Rect;       { Used to force a window onto the screen.
13 2781 --                                   If the window is not in the GrayRgn,
13 2782 --                                   then at least one corner of the window
13 2783 --                                   must lie within gStdWScreenRect. }
13 2784 --
13 2785 --      gSysWindowActive:  BOOLEAN;    { TRUE if the front window is a system window }
13 2786 --      gSystemStyle:     TextStyle;  { System's default text style }
13 2787 --      gTarget:          TEvtHandler; { the TEvtHandler that gets the first
13 2788 --                                   chance at DoCommand, DoSetupMenu,
13 2789 --                                   DoKeyCommand, Idle; should never be NIL --
13 2790 --                                   If you do not want your own target set this
13 2791 --                                   to the application object }
13 2792 --      { gTempRgn is a temporary RgnHandle created in InitToolbox.
13 2793 --        When debugging: before using gTempRgn, call UseTempRgn and when
13 2794 --        done call DoneWithTempRgn; this ensures that 2 routines do not try to
13 2795 --        use gTempRgn and the same time }
13 2796 --      gTempRgn:          RgnHandle;
13 2797 --      ($IFC qTrace)
13 2798 --      gTraceIdle:        BOOLEAN;
13 2799 --      ($ENDC)
13 2800 --      gUndoState:        BOOLEAN;
13 2801 --      gUndoCmd:          CmdNumber;
13 2802 --      ($IFC qDebug)
13 2803 --      gUsedBy:           Str255;      { The routine using gTempRgn }
13 2804 --      ($ENDC)
13 2805 --      gVarClipPicSize:   BOOLEAN;    { if TRUE, Pictures in the Clipboard
13 2806 --                                   are treated as variable size, depending on
13 2807 --                                   the window size; if FALSE (default), then
13 2808 --                                   pictures in the Clipboard are drawn and
13 2809 --                                   pasted ??? actual size }
13 2810 --      gWorkPort:         GrafPtr;     { Pointer to a graf port that is created
13 2811 --                                   during MacApp initialization }
13 2812 --      gWResSignature:    IDType;     { Signature used by TView.WRes }
13 2813 --      gWResType:         Str255;     { Type used by TView.WRes }
13 2814 --      gZeroRect:         Rect;       { SetRect(gZeroRect, 0, 0, 0, 0); }
13 2815 --      gZeroVPt:          VPoint;     { SetVPt(gZeroVPt, 0, 0); }
13 2816 --      gZeroVRect:        VRect;      { SetVRect(gZeroVRect, 0, 0, 0, 0); }
13 2817 --
13 2818 --
13 2819 --      +-----+
13 2820 --      | Global Procedures |
13 2821 --      +-----+
13 2822 --
13 2823 --      { D E B U G G I N G }
13 2824 --
13 2825 --      ($IFC qDebug)
13 2826 --      PROCEDURE DoneWithTempRgn;
13 2827 --      { Indicates that gTempRgn is no longer in use. Call this only if qDebug
13 2828 --        is true. }
13 2829 --
13 2830 --      PROCEDURE EntDebugger(entering: BOOLEAN);
13 2831 --
13 2832 --      FUNCTION LookupSymbol(VAR sym: String8): LONGINT;
13 2833 --
13 2834 --      PROCEDURE NotYetImplemented(where: Str255);
13 2835 --      { Put up an alert saying Not Yet Implemented and WriteLn where. }
13 2836 --
13 2837 --      PROCEDURE UseTempRgn(byWhom: Str255);
13 2838 --      { Call this when you are about to use gTempRgn and qDebug is true. Used
13 2839 --        with DoneWithTempRgn will prevent you from trying to use gTempRgn
13 2840 --        from two places at the same time. }
13 2841 --      ($ENDC qDebug)
13 2842 --
13 2843 --
13 2844 --
13 2845 --      { W I N D O W S }
13 2846 --
13 2847 --      PROCEDURE FreeWmgrWindow(w: WindowPtr; dispose: BOOLEAN);
13 2848 --      { Calls DisposeWindow if dispose is true, else calls CloseWindow. }
13 2849 --
13 2850 --      ($IFC qProceduralViews)
13 2851 --      FUNCTION NewPaletteWindow(itsRsrcID: INTEGER;
13 2852 --        wantHScrollBar: wantVScrollBar: BOOLEAN; itsDocument: TDocument;
13 2853 --        itsMainView: TView;
13 2854 --        itsPaletteView: TView; sizePalette: INTEGER; whichWay: VHSelect): TWindow;
13 2855 --      { Utility for creating MacDraw-line windows with 1 non-scrolling palette
13 2856 --        along the left edge (whichWay = h), or a non-scrolling status area at
13 2857 --        the top of the window (whichWay = v), and a main view that may or may
13 2858 --        not scroll. Signals Failure if the window could not be created. }
13 2859 --
13 2860 --      FUNCTION NewSimpleWindow(itsRsrcID: INTEGER; wantHScrollBar, wantVScrollBar: BOOLEAN;
13 2861 --        itsDocument: TDocument; itsView: TView): TWindow;
13 2862 --      { Utility for creating simple windows that contain 1 view and may or may not
13 2863 --        scroll. Signals Failure if the window could not be created. }
13 2864 --
13 2865 --      FUNCTION NewTWindow(itsRsrcID: INTEGER; itsDocument: TDocument): TWindow;
13 2866 --      { Utility for creating a window object. Used within NewPaletteWindow &
13 2867 --        NewSimpleWindow. Signals Failure if the window could not be created. }
13 2868 --      ($ENDC)
13 2869 --
13 2870 --      ($IFC qTemplateViews)

```

```

13 2871 -- FUNCTION NewTemplateWindow (viewRsrcID: INTEGER; itsDocument: TDocument): TWindow;
13 2872 -- { Creates a window using a 'WIND' resource and a 'view' template. }
13 2873 -- { $ENDC }
13 2874 --
13 2875 -- FUNCTION ParseTitleTemplate (VAR template: Str255;
13 2876 -- VAR preDocname, constTitle: INTEGER): BOOLEAN;
13 2877 -- { Used in TWindow.IWindow to parse the template for window titles. Returns
13 2878 -- TRUE if it changed the template. preDocname is start of document name
13 2879 -- in string; constChars is total # characters of window title that are
13 2880 -- constants. (These values are fixed regardless of the document name.)
13 2881 -- If preDocname is 0, then the entire title is constant. }
13 2882 --
13 2883 -- FUNCTION SubstituteInTitle (VAR title: Str255; newStuff: Str255;
13 2884 -- preDocname, constTitle: INTEGER): BOOLEAN;
13 2885 -- { Substitutes newStuff into the template as parsed by ParseTitleTemplate;
13 2886 -- returns TRUE if a substitution was made. }
13 2887 --
13 2888 --
13 2889 -- { C L I P B O A R D }
13 2890 --
13 2891 -- PROCEDURE CanPaste (aClipType: ResType);
13 2892 -- { Call this in your SetUpMenus code to register an ability to paste a
13 2893 -- particular type of Clipboard data }
13 2894 --
13 2895 -- FUNCTION PutDeskScrapData (aResType: ResType; aDataHandle: Handle): OSerr;
13 2896 -- { Call this from your TCommand method 'WriteToDeskScrap' (for a Command which
13 2897 -- changes the Clipboard) to write out data to the actual Desk Scrap. The
13 2898 -- return code from the Scrap Manager is returned as the function value --
13 2899 -- will be noErr unless something went wrong. This procedure leaves
13 2900 -- aDataHandle UNLOCKED. Rather than calling this, you can call the
13 2901 -- Toolbox routine PutScrap yourself }
13 2902 --
13 2903 --
13 2904 -- { E R R O R S }
13 2905 -- PROCEDURE ErrorAlert (err: OSerr; message: LONGINT);
13 2906 -- { Displays an error alert box. The message displayed in the alert is
13 2907 -- constructed base upon err and message, as described in the "Failure
13 2908 -- Handling" section of the "Cookbook" chapter of the MacApp manual. }
13 2909 --
13 2910 -- FUNCTION LookupErrString (value: INTEGER; resourceID: INTEGER;
13 2911 -- VAR str: Str255): BOOLEAN;
13 2912 -- { Looks up an error value in an 'errs' resource and returns TRUE
13 2913 -- if found. resourceID should be that of a MacApp errs resource;
13 2914 -- we do the lookup first in errAppTable+resourceID, which applications
13 2915 -- can use to override or extend the MacApp table. If the value is not
13 2916 -- found, str will be empty. }
13 2917 --
13 2918 -- FUNCTION MacAppAlert (alertID: INTEGER; filterProc: ProcPtr): INTEGER;
13 2919 -- { Does SetCursor (arrow) before calling Alert. }
13 2920 --
13 2921 -- PROCEDURE StdAlert (alertID: INTEGER);
13 2922 -- { no filterProc, reply ignored }
13 2923 --
13 2924 --
13 2925 -- { V I E W R E S O U R C E U T I L I T I E S }
13 2926 --
13 2927 -- FUNCTION NewViewRsrc (VAR p: UNIV Ptr): ViewRsrcHndl;
13 2928 -- { Convenience routine to create a new view resource template handle. Passes back
13 2929 -- in "p" the pointer to the first template entry. }
13 2930 --
13 2931 -- PROCEDURE DoneViewRsrc (viewRsrc: UNIV Handle; lastPtr: UNIV LONGINT);
13 2932 -- { Cuts back the handle to the proper length, based on lastPtr which should be the
13 2933 -- address of the next template entry. }
13 2934 --
13 2935 -- FUNCTION ExpandPtr (viewRsrc: UNIV Handle; VAR p: UNIV LONGINT;
13 2936 -- offset: LONGINT): Ptr;
13 2937 -- { Bumps size of view resource if necessary to add a new template. Amount to bump is set
13 2938 -- to at least kViewRsrcExpandAmt over the current size. "p" is offset to point to the
13 2939 -- next available position for a new template.
13 2940 -- Returns the old value of p before it was updated. }
13 2941 --
13 2942 -- FUNCTION ExpandPtrWStr (viewRsrc: UNIV Handle; VAR p: UNIV LONGINT;
13 2943 -- offset, len: LONGINT): Ptr;
13 2944 -- { Bumps size of view resource if necessary to add a new template which ends with a
13 2945 -- variable length string, the length of which is passed in len.
13 2946 -- Returns the old value of p before it was updated. }
13 2947 --
13 2948 -- PROCEDURE OffsetPtr (VAR p: UNIV LONGINT; offset: LONGINT);
13 2949 -- { Adds offset to p, forcing word alignment. }
13 2950 --
13 2951 -- PROCEDURE OffsetPtrWStr (VAR p: UNIV LONGINT; offset: LONGINT);
13 2952 -- { Adds offset to p, taking length of trailing variable length string into account,
13 2953 -- forcing word alignment. }
13 2954 --
13 2955 --
13 2956 -- { M I S C E L L A N E O U S }
13 2957 --
13 2958 -- PROCEDURE ExitMacApp;
13 2959 -- { Call this if for some reason you want to immediately exit the application. It calls
13 2960 -- gApplication.Terminate, cleans up some other internal stuff, and then calls ExitToShell.
13 2961 -- (Normally, you would not call this, because MacApp takes care of terminating
13 2962 -- the application.)
13 2963 --
13 2964 -- You must have first called InitToolbox before calling this. }
13 2965 --
13 2966 -- PROCEDURE InitToolbox (callsToMoreMasters: INTEGER);
13 2967 -- { Call this the very first thing in your main program. Does the essential
13 2968 -- Toolbox initialization; If you also use the printing unit UPrinting,
13 2969 -- call InitPrinting just after you call InitToolbox }

```

```
13 2970 --
13 2971 -- FUNCTION MakeNewRgn: RgnHandle;
13 2972 --   { Calls NewRgn, then FailNIL }
13 2973 --
13 2974 -- (* Removed because it has bugs and may not be the right solution
13 2975 -- PROCEDURE PrepareForDialog;
13 2976 --   { Deactivates the Front Window, in preparation for a coming Modal Dialog }
13 2977 -- *)
13 2978 --
13 2979 -- FUNCTION RectIsVisible(r: Rect): BOOLEAN;
13 2980 --   { Determine if a rect is visible in the current grafport. }
13 2981 --
13 2982 -- PROCEDURE SetHLPenState(fromHL, toHL: HLState);
13 2983 --   { Set the pen state for highlighting by XOR, given the highlight transition. You can use
13 2984 --     this if you use Paint<X> or Frame<X> or line drawing for your highlighting. }
13 2985 --   { ??? Add a pattern parameter or make a helper routine that has a pattern parameter ??? }
13 2986 --
13 2987 -- {SIFC qTemplateViews}
13 2988 -- PROCEDURE RegisterType (typeName: Str255; protoObj: TObj);
13 2989 --   { Register the given class name with an object of that class. }
13 2990 --
13 2991 -- PROCEDURE RegisterStdType (typeName: Str255; signature: IDType; protoObj: TObj);
13 2992 --   { Register the given class name and 'view' signature with an object of that class. }
13 2993 --
13 2994 -- FUNCTION FindStdView (signature: IDType): TView;
13 2995 --   { Returns the prototype view for the given signature. }
13 2996 -- {SENDC}
13 2997 --
13 2998 --
13 2999 -- IMPLEMENTATION
13 3000 --
13 3001 -- {$I UMacApp.Globals.p}
```

```

14 1 --      {$P}
14 2 --      { UMacApp.Globals.p }
14 3 --      { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
14 4 --
14 5 --      {-----+
14 6 --      | Local Constants      |
14 7 --      +-----}
14 8 --      CONST
14 9 --
14 10 --          kMaxIdleTime = MAXLONGINT;          { Default tick count sent to WaitNextEvent }
14 11 --
14 12 --          { Numbers to pass to SystemEdit }
14 13 --          kSysUndo      = 0;
14 14 --          kSysCut       = 2;
14 15 --          kSysCopy      = 3;
14 16 --          kSysPaste     = 4;
14 17 --          kSysClear     = 5;
14 18 --      {$IFC qWWNeeded}
14 19 --          kDbgWWHeight = 100;
14 20 --      {$ENDC}
14 21 --
14 22 --      {$IFC qDebug}
14 23 --          kRsrcCheckInterval = 100;
14 24 --      {$ENDC}
14 25 --
14 26 --
14 27 --      {-----+
14 28 --      | Local Types          |
14 29 --      +-----}
14 30 --      TYPE
14 31 --          dbugParams = RECORD                      { Format of 'debug' resource }
14 32 --                      boundsRect:  Rect;          { Rect of debugging window }
14 33 --                      fontNumber:  INTEGER;        { Font rsrc ID }
14 34 --                      fontSize:    INTEGER;        { Font size }
14 35 --                      numLines:    INTEGER;        { Number of lines }
14 36 --                      lineWidth:   INTEGER;        { Line width }
14 37 --                      title:       Str255;        { Actually, variable length }
14 38 --          END;
14 39 --          dbugParamsPtr = ^dbugParams;
14 40 --          dbugParamsHndl = ^dbugParamsPtr;
14 41 --
14 42 --      {-----+
14 43 --      | Local Variables      |
14 44 --      +-----}
14 45 --      VAR
14 46 --          { MacApp private global variables, in alphabetical order}
14 47 --          gCopyright:  StringHandle;
14 48 --          gETSPatch:   TrapPatch;    { patch for ExitToShell }
14 49 --
14 50 --
14 51 --      (*****
14 52 --      (*      External and Forward declared procedures
14 53 --      (*****
14 54 -- A      PROCEDURE Init2; FORWARD;
14 55 -- A      PROCEDURE Init3; FORWARD;
14 56 --
14 57 --      {$IFC qInspector}
14 58 -- A      PROCEDURE InitUInspector; EXTERNAL;
14 59 -- A      PROCEDURE MakeInspector; EXTERNAL;
14 60 -- A      PROCEDURE MakeInspectorWindow; EXTERNAL;
14 61 -- A      PROCEDURE AddObjectToInspector (theObject: TObjet); EXTERNAL;
14 62 --      {$ENDC}
14 63 --
14 64 --
14 65 --      (*****
14 66 --      (*      Regular procedures (in alphabetical order)
14 67 --      (*****
14 68 --
14 69 --      {$S Mares}
14 70 --      {-----+
14 71 --      | CanPaste              |
14 72 --      +-----}
14 73 -- A      PROCEDURE CanPaste(aClipType: ResType);
14 74 -- A      BEGIN
14 75 --          IF gClipView <> NIL THEN
14 76 --              IF gClipView.ContainsClipType(aClipType) THEN
14 77 --                  BEGIN
14 78 --                      gGotClipType := TRUE;
14 79 --                      gPrefClipType := aClipType;
14 80 --                  END;
14 81 -- A      END;
14 82 --
14 83 --      {$S Main}
14 84 --      {-----+
14 85 --      | CleanupMacApp        |
14 86 --      +-----}
14 87 -- A      PROCEDURE CleanupMacApp;
14 88 --      VAR
14 89 --          dontCare:  BOOLEAN;
14 90 -- A      BEGIN
14 91 --          SetupA5;    { ***** Called from trap patches *****}
14 92 --
14 93 --          UnpatchTrap(gETSPatch);
14 94 --
14 95 --          IF gApplication <> NIL THEN
14 96 --              gApplication.Terminate;
14 97 --
14 98 --          BusyRemove;

```

```

14 99 --
14 100 --      UnpatchAll;
14 101 --
14 102 --      {SIFC qWWNeeded}
14 103 --      IF WWRedirect(0, '') <> noErr THEN { ??? }; { close redirect file }
14 104 --      {ENDC}
14 105 --
14 106 --      {SIFC qTrace}
14 107 --      TRCTerminate;
14 108 --      {ENDC}
14 109 --
14 110 --      SetResLoad(TRUE);
14 111 --
14 112 --      dontCare := SetChooserAlert(gOldChooserFlag);
14 113 --
14 114 --      RestoreA5;
14 115 -0 A  END;
14 116 --
14 117 --      {SIFC qDebug}
14 118 --      {$S MADEBUG}
14 119 --      { See comment for UseTempRgn above. }
14 120 --      {SIFC qTrace}($D+){ENDC}
14 121 --      {-----+
14 122 --      | DoneWithTempRgn                                |
14 123 --      +-----}
14 124 -- A  PROCEDURE DoneWithTempRgn;
14 125 --      { Indicates that gTempRgn is no longer in use. Call this only if qDebug
14 126 --      is true. }
14 127 --
14 128 0- A  BEGIN
14 129 --      IF NOT gBusyTempRgn THEN
14 130 --          ProgramBreak('DoneWithTempRgn called, but gTempRgn is not locked');
14 131 --          gBusyTempRgn := FALSE;
14 132 --          gUsedBy := '';
14 133 --          SetEmptyRgn(gTempRgn);
14 134 -0 A  END { DoneWithTempRgn };
14 135 --      {SIFC qTrace}($D+){ENDC}
14 136 --      {ENDC}
14 137 --
14 138 --      {SIFC qDebug}
14 139 --      {$S MADEBUG}
14 140 --      {SIFC qTrace}($D+){ENDC}
14 141 --      {-----+
14 142 --      | EntDebugger                                    |
14 143 --      +-----}
14 144 -- A  PROCEDURE EntDebugger(entering: BOOLEAN);
14 145 0- A  BEGIN
14 146 --      BusyActivate(NOT entering);
14 147 -0 A  END;
14 148 --      {SIFC qTrace}($D+){ENDC}
14 149 --      {ENDC}
14 150 --
14 151 --      {$S MAError}
14 152 --      {-----+
14 153 --      | ErrorAlert                                    |
14 154 --      +-----}
14 155 -- A  PROCEDURE ErrorAlert(err: OSErr; message: LONGINT);
14 156 --      CONST
14 157 --          kMsgCmdErr      =      msgCmdErr DIV $10000;
14 158 --          kMsgAlert       =      msgAlert DIV $10000;
14 159 --          kMsgLookup      =      msgLookup DIV $10000;
14 160 --          kMsgAltRecov    =      msgAltRecovery DIV $10000;
14 161 --
14 162 --      TYPE
14 163 --          Converter = RECORD
14 164 --              CASE BOOLEAN OF
14 165 --                  TRUE: (message: LONGINT);
14 166 --                  FALSE: (hiWd, loWd: INTEGER);
14 167 --              END;
14 168 --
14 169 --      VAR c: Converter;
14 170 --          alertID: INTEGER;
14 171 --          genericAlert: BOOLEAN;
14 172 --          opString: Str255;
14 173 --          errStr: Str255;
14 174 --          recovErr: OSErr;
14 175 --          recovery: Str255;
14 176 --          x: BOOLEAN;
14 177 --
14 178 0- A  BEGIN
14 179 --      c.message := message;
14 180 --
14 181 --      alertID := phGenError; { the default alert }
14 182 --      genericAlert := TRUE;
14 183 --      opString := '';
14 184 --
14 185 1-  CASE c.hiWd OF
14 186 --      kMsgCmdErr:
14 187 2-  BEGIN
14 188 --          alertID := phCmdErr;
14 189 --          CmdToName(c.loWd, opString);
14 190 -2  END;
14 191 --      kMsgAlert:
14 192 2-  BEGIN
14 193 --          alertID := c.loWd;
14 194 --          genericAlert := FALSE;
14 195 -2  END;
14 196 --      kMsgLookup, kMsgAltRecov:
14 197 2-  BEGIN

```

```

14 198 --      x := LookupErrString(c.loWd, errOperationsID, opString);
14 199 --      END;
14 200 --      OTHERWISE
14 201 --      BEGIN
14 202 --          GetIndString(opString, c.hiWd, c.loWd);
14 203 --      END;
14 204 --      END;
14 205 --
14 206 --      IF genericAlert THEN
14 207 --      BEGIN
14 208 --          x := LookupErrString(err, errReasonID, errStr);
14 209 --
14 210 --          IF c.hiWd = kMsgAltRecov THEN
14 211 --              recovErr := c.loWd
14 212 --          ELSE
14 213 --              recovErr := err;
14 214 --
14 215 --          x := LookupErrString(recovErr, errRecoveryID, recovery);
14 216 --
14 217 --          ParamText(errStr, recovery, opString, gErrorParm3);
14 218 --
14 219 --          IF opString = '' THEN
14 220 --              alertID := phUnknownErr;
14 221 --          END;
14 222 --
14 223 --          StdAlert(alertID);
14 224 --
14 225 --          IF genericAlert THEN
14 226 --              ResetAlrtStage;
14 227 --      END;
14 228 --
14 229 --      {$S MAMain}
14 230 --      {-----+
14 231 --      |   ExitMacApp
14 232 --      +-----+}
14 233 --      A PROCEDURE ExitMacApp;
14 234 --      BEGIN
14 235 --          CleanupMacApp;
14 236 --          ExitToShell;
14 237 --      END;
14 238 --
14 239 --      {$S MAFinder}
14 240 --      { This is a dummy procedure to allow us to find the Finder segment }
14 241 --      {-----+
14 242 --      |   FinderSegProc
14 243 --      +-----+}
14 244 --      A PROCEDURE FinderSegProc;
14 245 --      BEGIN
14 246 --      END;
14 247 --
14 248 --      {$IFC qTemplateViews}
14 249 --      {$S MAOpen}
14 250 --      {-----+
14 251 --      |   FindStdView
14 252 --      +-----+}
14 253 --      A FUNCTION FindStdView (signature: IDType): TView;
14 254 --
14 255 --      VAR
14 256 --          i:      INTEGER;
14 257 --
14 258 --      BEGIN
14 259 --          FOR i := 1 TO gSignatureCount DO
14 260 --              IF LONGINT(gSignatures[i]) = LONGINT(signature) THEN
14 261 --              BEGIN
14 262 --                  FindStdView := TView(gSignatureViews[i]);
14 263 --                  EXIT(FindStdView);
14 264 --              END;
14 265 --          FindStdView := NIL;
14 266 --      END;
14 267 --      {$ENDC}
14 268 --
14 269 --      {$S MAMain}
14 270 --      {-----+
14 271 --      |   FreeWmgrWindow
14 272 --      +-----+}
14 273 --      A PROCEDURE FreeWmgrWindow(w: WindowPtr; dispose: BOOLEAN);
14 274 --          VAR theItemList:  Handle;
14 275 --      BEGIN
14 276 --          IF dispose THEN
14 277 --              DisposeWindow(w)
14 278 --          ELSE
14 279 --              CloseWindow(w);
14 280 --      END;
14 281 --
14 282 --      {$S MAREs}
14 283 --      {-----+
14 284 --      |   GetFocus
14 285 --      +-----+}
14 286 --      A PROCEDURE GetFocus;
14 287 --
14 288 --      BEGIN
14 289 --          GetPort(gSavePort);
14 290 --          gSaveOrg := thePort^.portRect.topLeft;
14 291 --          gSaveLongOffset := gLongOffset;
14 292 --          GetClip(gSaveClip);
14 293 --          gSaveFocusedView := gFocusedView;
14 294 --      END { GetFocus };
14 295 --
14 296 --      {$S Main} .      { Must be in a resident segment to ensure that UnloadAllSegments doesn't unload it. }

```





```

14 396 --      gFieldToStrRtn := @StdFieldToString;      { Get this set up early.      }
14 397 --      ($ENDC)
14 398 --
14 399 --      DefineConfiguration(gConfiguration);
14 400 --      ($IFC qNeedsROM128K)
14 401 --      { If we require 128K ROMs then make sure we are in fact running on them,
14 402 --        and if we are not, display an alert and exit the app }
14 403 --
14 404 --      IF NOT gConfiguration.hasROM128K THEN
14 405 1-      BEGIN
14 406 --          StdAlert(phUnsupportedROMs);
14 407 --          ExitToShell;
14 408 -1      END;
14 409 --      ($ENDC)
14 410 --
14 411 --      InitUPatch;
14 412 --      InitUFailure;
14 413 --
14 414 --      SetRGBColor(gRGBBlack, 0, 0, 0);      { Needs to come before InitUIInspector }
14 415 --      SetRGBColor(gRGBWhite, $FFFF, $FFFF, $FFFF);
14 416 --
14 417 --      ($IFC qInspector)
14 418 --          InitUIInspector;
14 419 --      ($ENDC)
14 420 --
14 421 --      ($IFC FALSE)
14 422 --          IF gConfiguration.hasROM128K THEN
14 423 --              SetFractEnable(TRUE);
14 424 --      ($ENDC)
14 425 --
14 426 --      ($IFC qDebug)
14 427 --          BuildObjNameTable;
14 428 --          gExperimenting := FALSE;
14 429 --          gDebugPrinting := FALSE;
14 430 --          gReportMenuChoices := FALSE;
14 431 --          gIntenseDebugging := FALSE;
14 432 --          gReportEvt := FALSE;
14 433 --          gMastReport := FALSE;
14 434 --          gRsrcReport := FALSE;
14 435 --          gMemMgtBreak := FALSE;
14 436 --      ($ENDC)
14 437 --
14 438 --      ($IFC qWWNeeded)
14 439 --          dParams := GetResource('dbug', kDebugParamsID);
14 440 --          IF dParams <> NIL THEN
14 441 --              WITH dbugParamsHndl(dParams)^^ DO
14 442 --                  WWInit(numLines, lineWidth)
14 443 --              ELSE
14 444 --                  WWInit(52, 80); { 52 lines of 80 characters }
14 445 --      ($ENDC)
14 446 --
14 447 --      ($IFC qDebug)
14 448 --          TRCInit;
14 449 --      ($ENDC)
14 450 --
14 451 --          SetInitHandler(@InitFailed);
14 452 --
14 453 --          { the following set up is necessary to call CleanupMacApp }
14 454 --          gApplication := NIL;
14 455 --
14 456 --          FailOsErr(TailPatch(gETSPatch, $A9F4, @CleanupMacApp));      { ExitToShell }
14 457 --
14 458 -0 A      END;
14 459 --
14 460 --      ($S MAINit) { Must be in the init segment; unloaded at start of event loop }
14 461 --      { Essential Toolbox initialization & other one-time initialization; part 2 }
14 462 --      ($IFC qTrace){$D+}($ENDC)
14 463 --      {-----+
14 464 --      |   Init3   |
14 465 --      +-----+}
14 466 -- A      PROCEDURE Init3;
14 467 --
14 468 --      CONST
14 469 --          MBarHeight =      $BAA;
14 470 --          SysFontSize =      $BA8;
14 471 --
14 472 --      VAR
14 473 --          message:      INTEGER;
14 474 --          aScrollBar:      TScrollBar;
14 475 --          aScroller:      TScroller;
14 476 --          aWindow:      TWindow;
14 477 --          aDeskScrapView:      TDeskScrapView;
14 478 --      ($IFC qWWNeeded)
14 479 --          outerRect:      Rect; { ??? }
14 480 --          dParams:      Handle;
14 481 --      ($ENDC)
14 482 --      ($IFC qDebug)
14 483 --          gDebugKeyMap:      KeyMap;      { the key state at start-up time }
14 484 --      ($ENDC)
14 485 --          fontSize:      INTEGER;
14 486 --
14 487 0- A      BEGIN
14 488 --
14 489 --          BusyInstall;
14 490 --
14 491 --      ($IFC qTrace)
14 492 --          gTraceSetupMenus := FALSE;
14 493 --          gTraceIdle := FALSE;
14 494 --          TRCInstall(Handle(gCodeSegs), Handle(gNonResidents),

```

```

14 495 --      @EntDebugger, @InspectObject, @LookupSymbol);
14 496 --
14 497 --      TRCFlag(@gAskAboutAlloc,      'A', 'ask about allocs');
14 498 --      TRCFlag(@gMemMgtBreak,        'B', 'mem mgt. break');
14 499 --      TRCFlag(@gReportMenuChoices,  'C', 'report menu commands');
14 500 --      TRCFlag(@gReportEvt,          'E', 'report events');
14 501 --      TRCFlag(@gAskFailure,         'F', 'ask about failures');
14 502 --      TRCFlag(@gIntenseDebugging,   'I', 'intense debugging');
14 503 --      TRCFlag(@gMastReport,         'M', 'report # masters');
14 504 --      TRCFlag(@gDebugPrinting,      'P', 'printing debug');
14 505 --      TRCFlag(@gRsrcReport,         'R', 'report rsrc usage');
14 506 --      TRCFlag(@gSegReport,          'S', 'report seg load');
14 507 --      TRCFlag(@gUnloadAllSegs,      'U', 'unload segments');
14 508 --      TRCFlag(@gExperimenting,      'X', 'experimenting');
14 509 --      ($ENDC qTrace)
14 510 --
14 511 --      gMainEventMask := everyEvent;
14 512 --
14 513 --      ($IFC qWWNeeded)
14 514 --      dParams := GetResource('dbug', kDebugParamsID);
14 515 --      IF dParams <> NIL THEN
14 516 --      BEGIN
14 517 --          HNoPurge(dParams);
14 518 --          WITH dbugParamsHndl(dParams)^^ DO
14 519 --              WWNew(boundsRect, title, TRUE, TRUE, fontNumber, fontSize);
14 520 --          HPurge(dParams);
14 521 --      END
14 522 --      ELSE
14 523 --      BEGIN
14 524 --          outerRect := screenBits.bounds;
14 525 --          InsetRect(outerRect, 5, 5);
14 526 --          outerRect.top := outerRect.bottom - kDbgWWHeight;
14 527 --          WWNew(outerRect, 'Debug Window', TRUE, TRUE, kDebugFont, kDebugSize);
14 528 --      END;
14 529 --      ($ENDC)
14 530 --
14 531 --      ($IFC qTrace)
14 532 --      TRCEnable(TRUE);
14 533 --      ($ENDC)
14 534 --
14 535 --      gCopyright := NewString(kCopyright);
14 536 --
14 537 --      ($IFC qDebug)
14 538 --      gBoolString[TRUE] := 'TRUE';
14 539 --      gBoolString[FALSE] := 'FALSE';
14 540 --
14 541 --      gRsrcCheck := kRsrcCheckInterval;
14 542 --      ($ENDC)
14 543 --
14 544 --      InitUMenuSetup;
14 545 --
14 546 --      gOldChooserFlag := SetChooserAlert(FALSE);
14 547 --
14 548 --      ($IFC qDebug)
14 549 --      GetKeys(gDebugKeyMap);
14 550 --      IF gDebugKeyMap[55] AND gDebugKeyMap[58] THEN { clover-option }
14 551 --          ProgramBreak('At start of application');
14 552 --      ($ENDC)
14 553 --
14 554 --      ($IFC qDebug)
14 555 --      IF cUndo - cEditBase <> kSysUndo THEN
14 556 --          WriteLn('Invalid UNDO command number');
14 557 --      IF cCut - cEditBase <> kSysCut THEN
14 558 --          WriteLn('Invalid CUT command number');
14 559 --      IF cCopy - cEditBase <> kSysCopy THEN
14 560 --          WriteLn('Invalid COPY command number');
14 561 --      IF cPaste - cEditBase <> kSysPaste THEN
14 562 --          WriteLn('Invalid PASTE command number');
14 563 --      IF cClear - cEditBase <> kSysClear THEN
14 564 --          WriteLn('Invalid CLEAR command number');
14 565 --      ($ENDC)
14 566 --
14 567 --      { Other 1-time initialization }
14 568 --      gTempRgn := MakeNewRgn;
14 569 --      gSaveClip := MakeNewRgn;
14 570 --
14 571 --      gClickCount := 0;
14 572 --      gLastUpTime := TickCount;
14 573 --      gLastClickPart := inDesk;
14 574 --      gIdlePhase := idleEnd;
14 575 --      gInBackground := FALSE; { When we start an app, it's in foreground }
14 576 --      gLastDeskAcc := gLastUpTime;
14 577 --      gIdleFreq := 0;
14 578 --      gLastIdle := 0;
14 579 --      gCursorRgn := MakeNewRgn;
14 580 --
14 581 --      { Create a work port for our convenience }
14 582 --      gWorkPort := @gFakeWindow;
14 583 --      IF gConfiguration.hasColorQD THEN
14 584 --          OpenCPort(CGrafPtr(gWorkPort))
14 585 --      ELSE
14 586 --          OpenPort(gWorkPort);
14 587 --      gFakeWindow.ControlList := NIL;
14 588 --
14 589 --      gNextSpaceMag := gLastUpTime;
14 590 --      gLowSpaceInterval := kLowSpaceInterval;
14 591 --
14 592 --      ($IFC qDebug)
14 593 --      gBusyTempRgn := FALSE;

```

```

14 594 --      gUsedBy := '';
14 595 --      {SENDC}
14 596 --
14 597 --      gNoChanges := Pointer(-1);
14 598 --
14 599 --      gStdHysteresis := Point($00040004); { ??? any better choice ??? }
14 600 --
14 601 --      gFrontWindow := NIL;
14 602 --
14 603 --      gZeroRect.topLeft := Point(0);
14 604 --      gZeroRect.botRight := Point(0);
14 605 --      SetVPt(gZeroVPt, 0, 0);
14 606 --      SetVRect(gZeroVRect, 0, 0, 0, 0);
14 607 --
14 608 --      ($IFC NOT qNeedsROM128K)
14 609 --      IF NOT gConfiguration.hasROM128K THEN
14 610 --          gMBarHeight := 20
14 611 --      ELSE
14 612 --      {SENDC}
14 613 --      IF gConfiguration.hasScriptManager THEN
14 614 --          gMBarHeight := GetMBarHeight
14 615 --      ELSE
14 616 --          gMBarHeight := PInteger(MBarHeight)^;
14 617 --
14 618 --      WITH screenBits.bounds DO
14 619 --      BEGIN
14 620 --          SetRect(gStdWMoveBounds, 4, gMBarHeight + 4, right - 4, bottom - 4);
14 621 --
14 622 --          { arbitrary minimum size; maximum size is screen size minus menu bar,
14 623 --            and half the title bar }
14 624 --          SetRect(gStdWSizeRect, 80, 80, right, bottom - gMBarHeight - 8);
14 625 --
14 626 --          SetRect(gStdWScreenRect, 16, gMBarHeight + 16, right - 16, bottom - 16);
14 627 --      END;
14 628 --
14 629 --      gOrthogonal[v] := h;
14 630 --      gOrthogonal[h] := v;
14 631 --
14 632 --      gPrinting := FALSE;
14 633 --      gCurrPrintHandler := NIL;
14 634 --      gDrawingPictScrap := FALSE;
14 635 --
14 636 --      gFinderPrinting := FALSE;
14 637 --      gCouldPrint := FALSE;
14 638 --
14 639 --      CountAppFiles(message, gFileCount);
14 640 --      gFinderPrinting := (message = appPrint);
14 641 --
14 642 --      gHeadCohandler := NIL;
14 643 --      gEventLevel := 0;
14 644 --
14 645 --      New(gNullPrintHandler);
14 646 --      FailNIL(gNullPrintHandler);
14 647 --      gNullPrintHandler.IPrintHandler(NIL);
14 648 --
14 649 --      gPrintHandler := gNullPrintHandler;
14 650 --
14 651 --      gFreeWindowList := NewList;
14 652 --
14 653 --      ($IFC qDebug)
14 654 --      gFreeWindowList.SetEltType('TWindow');
14 655 --      {SENDC}
14 656 --
14 657 --      gChooserOK := TRUE;
14 658 --
14 659 --      gClipWindow := NIL;
14 660 --
14 661 --      gGotClipType := FALSE;
14 662 --
14 663 --      gClipView := NIL;
14 664 --      gClipUndoView := NIL;
14 665 --
14 666 --      gNumUntitled := 1; { call the first document Untitled-1 }
14 667 --
14 668 --      gUndoState := kShowUndo;
14 669 --      gUndoCmd := cNoCommand;
14 670 --      gErrorParm3 := '';
14 671 --      gFocusedView := NIL;
14 672 --      gStdStaggerCount := 0;
14 673 --
14 674 --      gMBarDisplayed := kMBarDisplayed;
14 675 --      gMBarNotDisplayed := kMBarNotDisplayed;
14 676 --      gMBarHierarchical := kMBarHierarchical;
14 677 --
14 678 --      IF gConfiguration.hasScriptManager THEN
14 679 --          fontSize := GetDefFontSize
14 680 --      ELSE
14 681 --      ($IFC NOT qNeedsROM128K)
14 682 --      IF NOT gConfiguration.hasROM128K THEN
14 683 --          fontSize := 12
14 684 --      ELSE
14 685 --      {SENDC}
14 686 --          fontSize := PInteger(SysFontSize)^;
14 687 --          SetTextStyle(gSystemStyle, systemFont, [], fontSize, gRGBBlack);
14 688 --
14 689 --      ($IFC qTemplateViews)
14 690 --          gSignatureCount := 0;
14 691 --
14 692 --      NEW(gProtoList);      FailNIL(gProtoList);      gProtoList.IAssociation;

```

```

14 693 --      NEW(aWindow);      FailNIL(aWindow);      RegisterStdType('TWindow', 'wind', aWindow);
14 694 --      NEW(aScrollBar);    FailNIL(aScrollBar);    RegisterStdType('TScrollBar', 'sbar', aScrollBar);
14 695 --      NEW(aScroller);      FailNIL(aScroller);    RegisterStdType('TScroller', 'scri', aScroller);
14 696 --      NEW(aDeskScrapView); FailNIL(aDeskScrapView); RegisterType('TDeskScrapView', aDeskScrapView);
14 697 --      {$ENDC}
14 698 --
14 699 --      {$IFC qDebug}
14 700 --          IF gConfiguration.hasROM128K THEN Write('ROM128K ');
14 701 --          IF gConfiguration.hasHFS THEN Write('HFS Installed ');
14 702 --          IF gConfiguration.hasColorQD THEN Write('Has Color ');
14 703 --          IF gConfiguration.hasWaitNextEvent THEN Write('WaitNextEvent ');
14 704 --          Writeln;
14 705 --      {$ENDC}
14 706 --
14 707 --0 A      END;
14 708 --      {$IFC qTrace}{$D++}{$ENDC}
14 709 --
14 710 --      {$S MAError}
14 711 --      {-----+
14 712 --      | LookupErrString |
14 713 --      +-----}
14 714 -- A      FUNCTION LookupErrString(value: INTEGER; resourceID: INTEGER;
14 715 --                                VAR str: Str255): BOOLEAN;
14 716 --
14 717 --      {-----+
14 718 --      | SearchTable |
14 719 --      +-----}
14 720 -- B      FUNCTION SearchTable(value: INTEGER; resourceID: INTEGER;
14 721 --                                VAR str: Str255): BOOLEAN;
14 722 --
14 723 --          LABEL 1:
14 724 --              TYPE      PErrRecord = ^ErrRecord;
14 725 --                      ErrRecord = RECORD
14 726 --                          lowErr, highErr, index: INTEGER;
14 727 --                      END;
14 728 --
14 729 --              VAR table:      Handle;
14 730 --                  pEntry:      PErrRecord;
14 731 --                  tableOffset: LONGINT;
14 732 --                  lenTab:      INTEGER;
14 733 --                  strID:      INTEGER;
14 734 --                  i:          INTEGER;
14 735 --0 B      BEGIN
14 736 --          SearchTable := FALSE;
14 737 --          str := '';
14 738 --
14 739 --          table := GetResource('errs', resourceID);
14 740 --          IF table <> NIL THEN
14 741 --              BEGIN
14 742 --                  lenTab := GetHandleSize(Handle(table)) DIV SIZEOF(ErrRecord);
14 743 --
14 744 --                  strID := 0;
14 745 --                  tableOffset := 0;
14 746 --
14 747 --                  FOR i := 1 TO lenTab DO
14 748 --                      BEGIN
14 749 --                          pEntry := PErrRecord(Ord4(table) + tableOffset);
14 750 --
14 751 --                          WITH pEntry^ DO
14 752 --                              BEGIN
14 753 --                                  IF lowErr = 0 THEN
14 754 --                                      strID := index
14 755 --                                  ELSE IF (lowErr <= value) & (value <= highErr) THEN
14 756 --                                      BEGIN
14 757 --                                          IF index > 0 THEN
14 758 --                                              GetIndString(str, strID, index);
14 759 --                                              SearchTable := TRUE;
14 760 --                                              GOTO 1; { exit the loop }
14 761 --                                          END;
14 762 --                                      END;
14 763 --
14 764 --                                      tableOffset := tableOffset + SIZEOF(ErrRecord);
14 765 --                                  END;
14 766 --                              END;
14 767 --                          END;
14 768 --0 B      END;
14 769 --
14 770 --0 A      BEGIN
14 771 --          IF SearchTable(value, errAppTable+resourceId, str) THEN
14 772 --              LookupErrString := TRUE
14 773 --          ELSE
14 774 --              LookupErrString := SearchTable(value, resourceId, str);
14 775 --0 A      END;
14 776 --
14 777 --      {$IFC qDebug}
14 778 --      {$S MAError}
14 779 --      {$IFC qTrace}{$D++}{$ENDC}
14 780 --      {-----+
14 781 --      | LookupSymbol |
14 782 --      +-----}
14 783 -- A      FUNCTION LookupSymbol(VAR sym: String8): LONGINT;
14 784 --0 A      BEGIN
14 785 --          IF gInitialized THEN
14 786 --              LookupSymbol := gTarget.LookupSymbol(sym)
14 787 --          ELSE
14 788 --              LookupSymbol := -1;
14 789 --0 A      END;
14 790 --      {$IFC qTrace}{$D++}{$ENDC}
14 791 --      {$ENDC}

```

```

14 792 --
14 793 --      {$S MAMain}
14 794 --      {-----+
14 795 --      |   MacAppAlert
14 796 --      +-----+
14 797 -- A  FUNCTION MacAppAlert(alertID: INTEGER; filterProc: ProcPtr): INTEGER;
14 798 -- A  BEGIN
14 799 --      {$IFC qDebug}
14 800 --      gRsrcCheck := 0;                { force immediate check.      }
14 801 --      {$ENDC}
14 802 --
14 803 --      SetCursor(arrow);
14 804 --      MacAppAlert := Alert(alertID, filterProc);
14 805 --      SetEmptyRgn(gCursorRgn);        { Make sure cursor get reset.  }
14 806 -- A  END;
14 807 --
14 808 --
14 809 --      {$S MAMain}
14 810 --      {$IFC qTrace}{$D+}{$ENDC}
14 811 --      {-----+
14 812 --      |   MakeNewRgn
14 813 --      +-----+
14 814 -- A  FUNCTION MakeNewRgn: RgnHandle;
14 815 --      VAR
14 816 --          aRgn:      RgnHandle;
14 817 -- A  BEGIN
14 818 --          aRgn := NewRgn;
14 819 --          FailNIL(aRgn);
14 820 --          MakeNewRgn := aRgn;
14 821 -- A  END;
14 822 --      {$IFC qTrace}{$D+}{$ENDC}
14 823 --
14 824 --
14 825 --      {$IFC qProceduralViews}
14 826 --      {$S MAOpen}
14 827 --      {-----+
14 828 --      |   NewPaletteWindow
14 829 --      +-----+
14 830 -- A  FUNCTION NewPaletteWindow(itsRsrcID: INTEGER; wantHScrollBar, wantVScrollBar: BOOLEAN;
14 831 --                               itsDocument: TDocument; itsMainView: TView; itsPaletteView: TView;
14 832 --                               sizePalette: INTEGER; whichWay: VHSelect): TWindow;
14 833 --      VAR aWindow:      TWindow;
14 834 --          aScroller:    TScroller;
14 835 --          fi:            FailInfo;
14 836 --          itsSize:       VPoint;
14 837 --          itsLocation:   VPoint;
14 838 --          wSize:         Point;
14 839 --
14 840 --      {-----+
14 841 --      |   HdlnPWindow
14 842 --      +-----+
14 843 -- B  PROCEDURE HdlnPWindow(error: INTEGER; message: LONGINT);
14 844 -- B  BEGIN
14 845 --      aWindow.Free;
14 846 -- B  END;
14 847 --
14 848 -- A  BEGIN
14 849 --      aWindow := NewTWindow(itsRsrcID, itsDocument);
14 850 --
14 851 --      WITH aWindow.fResizeLimits.topLeft DO
14 852 --          vh[whichWay] := vh[whichWay] + sizePalette;
14 853 --
14 854 --      CatchFailures(fi, HdlnPWindow);
14 855 --
14 856 --      aWindow.AddSubview(itsPaletteView);
14 857 --
14 858 --      itsLocation := gZeroVpt;
14 859 --      itsLocation.vh[whichWay] := sizePalette;
14 860 --      IF wantHScrollBar OR wantVScrollBar THEN
14 861 --      BEGIN
14 862 --          SetVpt(itsSize, aWindow.fSize.h - kSBarSizeMinus1,
14 863 --                aWindow.fSize.v - kSBarSizeMinus1);
14 864 --          itsSize.vh[whichWay] := itsSize.vh[whichWay] - sizePalette;
14 865 --          New(aScroller);
14 866 --          FailNIL(aScroller);
14 867 --          aScroller.IScroller(aWindow, itsLocation, itsSize, sizeRelSuperView,
14 868 --                               sizeRelSuperView, 0, 0, wantHScrollBar, wantVScrollBar);
14 869 --          aScroller.AddSubview(itsMainView);
14 870 --      END
14 871 --      ELSE
14 872 --          aWindow.AddSubview(itsMainView);
14 873 --
14 874 --      aWindow.fTarget := itsMainView;
14 875 --
14 876 --      { make frames be the right size }
14 877 --      WITH aWindow.fWmgrWindow^.portRect DO
14 878 --      BEGIN
14 879 --          wSize := botRight;
14 880 --          {$H-}
14 881 --          SubPt(topLeft, wSize);
14 882 --          {$H+}
14 883 --      END;
14 884 --      aWindow.Resize(wSize.h, wSize.v, FALSE);
14 885 --
14 886 --      NewPaletteWindow := aWindow;
14 887 --
14 888 --      Success(fi);
14 889 -- A  END;
14 890 --      {$ENDC}

```

```

14 891 --
14 892 --
14 893 --      {$IFC qProceduralViews}
14 894 --      {$S MAOpen}
14 895 --      {-----+
14 896 --      |   NewSimpleWindow
14 897 --      +-----}
14 898 -- A  FUNCTION NewSimpleWindow(itsRsrcID: INTEGER; wantHScrollBar, wantVScrollBar: BOOLEAN;
14 899 --                                itsDocument: TDocument; itsView: TView): TWindow;
14 900 --      VAR aWindow:      TWindow;
14 901 --          aScroller:    TScroller;
14 902 --          fi:            FailInfo;
14 903 --          itsSize:       VPoint;
14 904 --          wSize:         Point;
14 905 --          sBarOffsets:   VRect;
14 906 --
14 907 --      {-----+
14 908 --      |   HdlnSWindow
14 909 --      +-----}
14 910 -- B  PROCEDURE HdlnSWindow(error: INTEGER; message: LONGINT);
14 911 -- B  BEGIN
14 912 --      aWindow.Free;
14 913 -- B  END;
14 914 --
14 915 -- A  BEGIN
14 916 --      aWindow := NewTWindow(itsRsrcID, itsDocument);
14 917 --
14 918 --      aScroller := NIL;
14 919 --
14 920 --      CatchFailures(fi, HdlnSWindow);
14 921 --
14 922 --      IF wantHScrollBar | wantVScrollBar THEN
14 923 --      BEGIN
14 924 --          sBarOffsets := gZeroVRect;
14 925 --          itsSize := aWindow.fSize;
14 926 --          IF wantHScrollBar THEN
14 927 --          BEGIN
14 928 --              itsSize.v := itsSize.v - kSBarSizeMinus1;
14 929 --              IF NOT wantVScrollBar THEN
14 930 --                  sBarOffsets.right := -kSBarSizeMinus1;
14 931 --          END;
14 932 --          IF wantVScrollBar THEN
14 933 --          BEGIN
14 934 --              itsSize.h := itsSize.h - kSBarSizeMinus1;
14 935 --              IF NOT wantHScrollBar THEN
14 936 --                  sBarOffsets.bottom := -kSBarSizeMinus1;
14 937 --          END;
14 938 --          New(aScroller);
14 939 --          FailNIL(aScroller);
14 940 --          aScroller.IScroller(aWindow, gZeroVpt, itsSize, sizeRelSuperview,
14 941 --                                sizeRelSuperview, 0, 0, wantHScrollBar, wantVScrollBar);
14 942 --          aScroller.fSBarOffsets := sBarOffsets;
14 943 --          IF itsView <> NIL THEN
14 944 --              aScroller.AddSubview(itsView);
14 945 --          END
14 946 --      ELSE
14 947 --          IF itsView <> NIL THEN
14 948 --              aWindow.AddSubview(itsView);
14 949 --          END
14 950 --      IF itsView <> NIL THEN
14 951 --          aWindow.fTarget := itsView;
14 952 --
14 953 --      { make sure window and subviews are the right size }
14 954 --      WITH aWindow.fWmgrWindow^.portRect DO
14 955 --      BEGIN
14 956 --          wSize := botRight;
14 957 --          {$H-}
14 958 --          SubPt(topLeft, wSize);
14 959 --          {$H+}
14 960 --      END;
14 961 --      aWindow.Resize(wSize.h, wSize.v, FALSE);
14 962 --
14 963 --      NewSimpleWindow := aWindow;
14 964 --
14 965 --      Success(fi);
14 966 -- A  END;
14 967 --      {$ENDC}
14 968 --
14 969 --
14 970 --      {$IFC qProceduralViews}
14 971 --      {$S MAOpen}
14 972 --      {-----+
14 973 --      |   NewTWindow
14 974 --      +-----}
14 975 -- A  FUNCTION NewTWindow(itsRsrcID: INTEGER; itsDocument: TDocument): TWindow;
14 976 --      VAR aWmgrWindow:   WindowPtr;
14 977 --          aWindow:       TWindow;
14 978 --          canResize:     BOOLEAN;
14 979 --          canClose:      BOOLEAN;
14 980 --          fi:             FailInfo;
14 981 --
14 982 --      {-----+
14 983 --      |   HdlnNewWObj
14 984 --      +-----}
14 985 -- B  PROCEDURE HdlnNewWObj(error: INTEGER; message: LONGINT);
14 986 -- B  BEGIN
14 987 --      { the wmgrWindow is known to exist }
14 988 --      { Since aWindow didn't get created, the wmgrWindow won't be
14 989 --        freed unless we do it here. }

```

```

14 990 --
14 991 --      FreeWmgrWindow(aWmgrWindow, TRUE);
14 992 --
14 993 -0 B      END;
14 994 --
14 995 0- A      BEGIN
14 996 --          aWmgrWindow := gApplication.GetRsrcWindow(NIL, itsRsrcID, canResize, canClose);
14 997 --          { GetRsrcWindow signals Failure }
14 998 --
14 999 --          CatchFailures(fi, Hd1NewWObj);
14 1000 --
14 1001 --          aWindow := NIL;
14 1002 --
14 1003 --          New(aWindow);
14 1004 --          FailNIL(aWindow);
14 1005 --          Success(fi);
14 1006 --
14 1007 --          aWindow.IWindow(itsDocument, aWmgrWindow, canResize, canClose, TRUE); { TRUE means can dispose wmgr window }
14 1008 --
14 1009 --          NewTWindow := aWindow;
14 1010 --
14 1011 -0 A      END;
14 1012 --          {$ENDC}
14 1013 --
14 1014 --
14 1015 --          {$IFC qTemplateViews}
14 1016 --          {$S MAOpen}
14 1017 --          {-----+
14 1018 --          |      NewTemplateWindow
14 1019 --          +-----}
14 1020 -- A      FUNCTION NewTemplateWindow (viewRsrcID: INTEGER; itsDocument: TDocument): TWindow;
14 1021 --
14 1022 --      VAR
14 1023 --          theWindow:      TWindow;
14 1024 --          theTarget:      TView;
14 1025 --
14 1026 0- A      BEGIN
14 1027 --          theWindow := TWindow(gTarget.DoCreateViews(itsDocument, NIL, viewRsrcID));
14 1028 --          IF theWindow <> NIL THEN
14 1029 1-      BEGIN
14 1030 --              IF theWindow.fWmgrWindow <> NIL THEN
14 1031 --                  WITH theWindow.fWmgrWindow^.portRect DO
14 1032 --                      theWindow.Resize(right - left, bottom - top, FALSE);
14 1033 --              IF theWindow.fTargetID <> kNoIdentifier THEN
14 1034 2-      BEGIN
14 1035 --                  theTarget := theWindow.FindSubView(theWindow.fTargetID);
14 1036 --                  IF theTarget <> NIL THEN
14 1037 --                      theWindow.fTarget := theTarget;
14 1038 -2      END;
14 1039 -1      END;
14 1040 --          NewTemplateWindow := theWindow;
14 1041 -0 A      END;
14 1042 --          {$ENDC}
14 1043 --
14 1044 --
14 1045 --          {$IFC qDebug}
14 1046 --          {$S MADebug}
14 1047 --          {$IFC qTrace}{$D+}{$ENDC}
14 1048 --          {-----+
14 1049 --          |      NotYetImplemented
14 1050 --          +-----}
14 1051 -- A      PROCEDURE NotYetImplemented(where: Str255);
14 1052 0- A      BEGIN
14 1053 --          StdAlert(phUnimplemented);
14 1054 --          WriteLn('Feature Not Yet Implemented: ', where);
14 1055 -0 A      END;
14 1056 --          {$IFC qTrace}{$D+}{$ENDC}
14 1057 --          {$ENDC}
14 1058 --
14 1059 --          {$S MARES}
14 1060 --          {-----+
14 1061 --          |      ParseTitleTemplate
14 1062 --          +-----}
14 1063 -- A      FUNCTION ParseTitleTemplate(VAR template: Str255; VAR preDocname, constTitle: INTEGER): BOOLEAN;
14 1064 --      CONST
14 1065 --          kPreDocname = '<<<';
14 1066 --          kPreSize = 3;
14 1067 --          kPostDocname = '>>>';
14 1068 --          kPostSize = 3;
14 1069 --
14 1070 --      VAR
14 1071 --          x: INTEGER;
14 1072 --
14 1073 -- B      FUNCTION FindPos (pattern: Str255; VAR source: Str255): INTEGER;
14 1074 --      VAR
14 1075 --          position: INTEGER;
14 1076 0- B      BEGIN
14 1077 --          IF gConfiguration.hasScriptManager THEN
14 1078 1-      REPEAT
14 1079 --              position := POS(pattern, source);
14 1080 -1      UNTIL (position = 0) OR (CharByte(@source, position) = 0)
14 1081 --      ELSE
14 1082 --          position := POS(pattern, source);
14 1083 --          FindPos := position;
14 1084 -0 B      END;
14 1085 --
14 1086 0- A      BEGIN
14 1087 --          IF template = '' THEN
14 1088 1-      BEGIN

```

```

14 1089 --      preDocname := 1;
14 1090 --      constTitle := 0;
14 1091 -1      END
14 1092 --      ELSE
14 1093 1-      BEGIN
14 1094 --      preDocname := FindPos(kPreDocname, template);
14 1095 --      IF preDocname > 0 THEN
14 1096 2-      BEGIN
14 1097 --      Delete(template, preDocname, kPreSize);
14 1098 --
14 1099 --      x := FindPos(kPostDocname, template);
14 1100 --      IF x = 0 THEN
14 1101 --      constTitle := preDocname - 1
14 1102 --      ELSE
14 1103 3-      BEGIN
14 1104 --      Delete(template, x, kPostSize);
14 1105 --      constTitle := Length(template) - x + preDocname;
14 1106 -3      END;
14 1107 -2      END;
14 1108 -1      END;
14 1109 --
14 1110 --      ParseTitleTemplate := preDocname > 0;
14 1111 -0 A  END;
14 1112 --
14 1113 --      {$S MAActivate}
14 1114 --      {-----+
14 1115 --      | PutDeskScrapData |
14 1116 --      +-----}
14 1117 -- A  FUNCTION PutDeskScrapData(aResType: ResType; aDataHandle: Handle): OSErr;
14 1118 --      VAR err: LONGINT;
14 1119 0- A  BEGIN
14 1120 --      MoveHHI(aDataHandle);
14 1121 --      HLock(aDataHandle);
14 1122 --      err := PutScrap(GetHandleSize(aDataHandle), aResType, aDataHandle^);
14 1123 --      HUnlock(aDataHandle);
14 1124 --      {$IFC qDebug}
14 1125 --      IF err <> noErr THEN
14 1126 --      WriteLn('Error from PutScrap is: ', err:1);
14 1127 --      {$ENDC}
14 1128 --      PutDeskScrapData := err;
14 1129 -0 A  END;
14 1130 --
14 1131 --      {$S Mares}
14 1132 --      {-----+
14 1133 --      | RectIsVisible |
14 1134 --      +-----}
14 1135 -- A  FUNCTION RectIsVisible(r: Rect): BOOLEAN;
14 1136 0- A  BEGIN
14 1137 --      {$IFC FALSE} { ??? Eventually get rid of this in MacApp 2.0? }
14 1138 --      IF gPrinting THEN
14 1139 --      RectIsVisible := gCurrPrintHandler.WouldPrintRect(r)
14 1140 --      ELSE
14 1141 --      {$ENDC}
14 1142 --      IF gDrawingPictScrap THEN
14 1143 --      RectIsVisible := TRUE
14 1144 --      ELSE
14 1145 --      RectIsVisible := RectInRgn(r, thePort^.visRgn);
14 1146 --      { ??? need to check the clipping also ??? }
14 1147 -0 A  END;
14 1148 --
14 1149 --
14 1150 --      {$IFC qTemplateViews}
14 1151 --      {$S MAMain}
14 1152 --      {-----+
14 1153 --      | RegisterType |
14 1154 --      +-----}
14 1155 -- A  PROCEDURE RegisterType (typeName: Str255; protoObj: TObj);
14 1156 --      VAR
14 1157 --      anEntry: TEntry;
14 1158 0- A  BEGIN
14 1159 --      gProtoList.InsertEntry(typeName, '');
14 1160 --      anEntry := gProtoList.EntryWithKey(typeName);
14 1161 --      anEntry.fValue := StringHandle(protoObj);
14 1162 -0 A  END;
14 1163 --      {$ENDC}
14 1164 --
14 1165 --
14 1166 --      {$IFC qTemplateViews}
14 1167 --      {$S MAMain}
14 1168 --      {-----+
14 1169 --      | RegisterStdType |
14 1170 --      +-----}
14 1171 -- A  PROCEDURE RegisterStdType (typeName: Str255; signature: IDType; protoObj: TObj);
14 1172 0- A  BEGIN
14 1173 --      gSignatureCount := gSignatureCount + 1;
14 1174 --      {$IFC qDebug}
14 1175 --      IF gSignatureCount >= kMaxSignatures THEN
14 1176 --      ProgramBreak('Maximum number of signature views exceeded.');
```



```

14 1188 -- {-----+
14 1189 -- |   SetFocus                               |
14 1190 -- +-----+
14 1191 -- A  PROCEDURE SetFocus;
14 1192 --
14 1193 0- A  BEGIN
14 1194 --      SetPort(gSavePort);
14 1195 --      SetOrigin(gSaveOrg.h, gSaveOrg.v);
14 1196 --      SetClip(gSaveClip);
14 1197 --      gLongOffset := gSaveLongOffset;
14 1198 --      gFocusedView := gSaveFocusedView;
14 1199 0- A  END ( SetFocus );
14 1200 --
14 1201 -- { $S MAREs }
14 1202 -- {-----+
14 1203 -- |   SetHLPenState                           |
14 1204 -- +-----+
14 1205 -- A  PROCEDURE SetHLPenState(fromHL, toHL: HLPenState);
14 1206 --      VAR pPat:   ^Pattern;
14 1207 --      mode:   INTEGER;
14 1208 0- A  BEGIN
14 1209 --      mode := patXOR;      ( every transition except h1On <-> h1Dim uses patXOR )
14 1210 --
14 1211 --
14 1212 --      IF fromHL = toHL THEN
14 1213 --          pPat := @white
14 1214 --
14 1215 --      ELSE IF fromHL + toHL = h1OffOn THEN
14 1216 --          pPat := @black
14 1217 --
14 1218 --      ELSE
14 1219 --          pPat := @gray; ( ??? make this pattern a parameter ??? )
14 1220 --
14 1221 --      IF fromHL + toHL = h1DimOn THEN
14 1222 --          mode := NOTpatXOR;
14 1223 --
14 1224 --      PenMode(mode);
14 1225 --      PenPat(pPat^);
14 1226 0- A  END;
14 1227 --
14 1228 -- { $S MAMain }
14 1229 -- {-----+
14 1230 -- |   StdAlert                               |
14 1231 -- +-----+
14 1232 -- A  PROCEDURE StdAlert(alertId: INTEGER);
14 1233 --      VAR reply: INTEGER;
14 1234 0- A  BEGIN
14 1235 --      reply := MacAppAlert(alertId, NIL);
14 1236 0- A  END;
14 1237 --
14 1238 -- { $S MAREs }
14 1239 -- {-----+
14 1240 -- |   SubstituteInTitle                       |
14 1241 -- +-----+
14 1242 -- A  FUNCTION SubstituteInTitle(VAR title: Str255; newStuff: Str255;
14 1243 --                                preDocname, constTitle: INTEGER): BOOLEAN;
14 1244 0- A  BEGIN
14 1245 --      IF preDocname > 0 THEN
14 1246 1-  BEGIN
14 1247 --          IF constTitle = 0 THEN
14 1248 --              title := newStuff
14 1249 --          ELSE
14 1250 2-  BEGIN
14 1251 --              Delete(title, preDocname, Length(title) - constTitle);
14 1252 --              Insert(newStuff, title, preDocname);
14 1253 2-  END;
14 1254 --              SubstituteInTitle := TRUE;
14 1255 1-  END
14 1256 --          ELSE
14 1257 --              SubstituteInTitle := FALSE;
14 1258 0- A  END;
14 1259 --
14 1260 -- { $IFC qDebug }
14 1261 -- { $S MADebug }
14 1262 -- { $IFC qTrace } { $D+ } { $ENDC }
14 1263 -- {-----+
14 1264 -- |   UseTempRgn                             |
14 1265 -- +-----+
14 1266 -- A  PROCEDURE UseTempRgn(byWhom: Str255);
14 1267 --      { Call this when you are about to use gTempRgn and qDebug is true.  Used
14 1268 --        with DoneWithTempRgn will prevent you from trying to use gTempRgn
14 1269 --        from two places at the same time. }
14 1270 --
14 1271 0- A  BEGIN
14 1272 --      IF gBusyTempRgn THEN
14 1273 1-  BEGIN
14 1274 --          WriteLn(' ', byWhom, ' is trying to lock gTempRgn, ');
14 1275 --          WriteLn('but it is already locked by ', gUsedBy, ' ');
14 1276 --          ProgramBreak('Error in UseTempRgn');
14 1277 1-  END
14 1278 --      ELSE
14 1279 1-  BEGIN
14 1280 --          gBusyTempRgn := TRUE;
14 1281 --          gUsedBy := byWhom;
14 1282 1-  END;
14 1283 0- A  END;
14 1284 -- { $IFC qTrace } { $D+ } { $ENDC }
14 1285 -- { $ENDC qDebug }
14 13002 -- { $I UMacApp.TEvtHandler.p }

```



```

15 1 --      {SP}
15 2 --      { UMacApp.TEvtHandler.p }
15 3 --      { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
15 4 --
15 5 --      (*****
15 6 --      (*      T E v t H a n d l e r      *)
15 7 --      (*****
15 8 --      {$S Mares}
15 9 --      {-----+
15 10 --      | IEvtHandler
15 11 --      +-----}
15 12 -- A  PROCEDURE TEvtHandler.IEvtHandler(itsNextHandler: TEvtHandler);
15 13 0- A  BEGIN
15 14 --      fNextHandler := itsNextHandler;
15 15 --      fIdleFreq := kMaxIdleTime;          { Assume it never wants idle time }
15 16 --      fLastIdle := 0;
15 17 -0 A  END;
15 18 --
15 19 --      {$S MAClose}
15 20 --      {-----+
15 21 --      | Free
15 22 --      +-----}
15 23 -- A  PROCEDURE TEvtHandler.Free; OVERRIDE;
15 24 0- A  BEGIN
15 25 --      {??? Should this call SetTarget to set the target?}
15 26 --      IF gTarget = SELF THEN
15 27 --          IF fNextHandler = NIL THEN
15 28 --              gTarget := gApplication
15 29 --          ELSE
15 30 --              gTarget := fNextHandler;
15 31 --      INHERITED Free;
15 32 -0 A  END;
15 33 --
15 34 --      {$IFC qDebug}
15 35 --      {$S MAFields}
15 36 --      {-----+
15 37 --      | Fields
15 38 --      +-----}
15 39 -- A  PROCEDURE TEvtHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
15 40 --                                fieldAddr: Ptr;
15 41 --                                fieldType: INTEGER));
15 42 0- A  BEGIN
15 43 --      DoToField('TEvtHandler', NIL, bClass);
15 44 --      DoToField('fNextHandler', @fNextHandler, bObject);
15 45 --      DoToField('fIdleFreq', @fIdleFreq, bLongInt);
15 46 --      DoToField('fLastIdle', @fLastIdle, bLongInt);
15 47 --      INHERITED Fields(DoToField);
15 48 -0 A  END (TEvtHandler.Fields);
15 49 --      {$ENDC}
15 50 --
15 51 --
15 52 --      {$IFC qTemplateViews}
15 53 --      {$S MAOpen}
15 54 --      {-----+
15 55 --      | CreateAView
15 56 --      +-----}
15 57 -- A  FUNCTION TEvtHandler.CreateAView ( itsDocument: TDocument; itsSuperView: TView;VAR itsParams: Ptr): TView;
15 58 --      VAR
15 59 --          anEntry: TEntry;
15 60 --          aView: TView;
15 61 --          newView: TView;
15 62 --
15 63 0- A  BEGIN
15 64 --      IF fNextHandler <> NIL THEN
15 65 --          CreateAView := fNextHandler.CreateAView(itsDocument, itsSuperView, itsParams)
15 66 --      ELSE
15 67 1-  BEGIN
15 68 --          WITH ViewTemplatePtr(itsParams)^ DO
15 69 --              IF LENGTH(itsType) = 0 THEN
15 70 --                  aView := FindStdView(itsSignature)
15 71 --              ELSE
15 72 2-  BEGIN
15 73 --                  anEntry := gProtoList.EntryWithKey(itsType);
15 74 --                  IF anEntry <> NIL THEN
15 75 --                      aView := TView(anEntry.fValue)
15 76 --                  ELSE
15 77 --                      aView := NIL;
15 78 -2  END;
15 79 --
15 80 --          IF aView <> NIL THEN
15 81 2-  BEGIN
15 82 --              newView := TView(aView.ShallowClone);
15 83 --              FailNIL(newView);
15 84 --              newView.IRes(itsDocument, itsSuperView, itsParams);
15 85 --              CreateAView := newView;
15 86 -2  END
15 87 --          ELSE
15 88 2-  BEGIN
15 89 --              {$IFC qDebug}
15 90 --                  ProgramBreak(CONCAT('Class ', ViewTemplatePtr(itsParams)^.itsType,
15 91 --                                     ' is not registered.));
15 92 --              {$ENDC}
15 93 --              CreateAView := NIL;
15 94 -2  END;
15 95 -1  END;
15 96 -0 A  END;
15 97 --      {$ENDC}
15 98 --

```

```

15 99 --
15 100 --      {$IFC qTemplateViews}
15 101 --      {$S MAOpen}
15 102 --      {-----+
15 103 --      | DoCreateViews
15 104 --      +-----}
15 105 -- A  FUNCTION TEvtHandler.DoCreateViews (itsDocument: TDocument; parentView: TView; itsRsrcID: INTEGER): TView;
15 106 --      VAR
15 107 --          i:          INTEGER;
15 108 --          numViews:   INTEGER;
15 109 --          aView:       TView;
15 110 --          aHandle:     ViewRsrcHndl;
15 111 --          theViewInfo: ViewTemplatePtr;
15 112 --          lastParentID: IDType;
15 113 --          lastParent:  TView;
15 114 --          firstView:   TView;
15 115 --          fi:          FailInfo;
15 116 --
15 117 -- B      PROCEDURE HdlDoCreateViews (error: OSerr; message: LONGINT);
15 118 -- 0- B      BEGIN
15 119 --          FreeObject(firstView);
15 120 -- 0 B      END;
15 121 --
15 122 -- A  BEGIN
15 123 --      IF fNextHandler <> NIL THEN
15 124 --          DoCreateViews := fNextHandler.DoCreateViews(itsDocument, parentView, itsRsrcID)
15 125 --      ELSE
15 126 -- 1-      BEGIN
15 127 --          firstView := NIL; { Assume the worst. }
15 128 --
15 129 --          aHandle := ViewRsrcHndl(GetResource('view', itsRsrcID));
15 130 --          FailNIL(Handle(aHandle));
15 131 --          MoveHHI(Handle(aHandle));
15 132 --          HLock(Handle(aHandle));
15 133 --
15 134 --          CatchFailures(fi, HdlDoCreateViews);
15 135 --
15 136 --          numViews := aHandle^.numViews;
15 137 --          theViewInfo := @aHandle^.theViews;
15 138 --          lastParentID := kNoIdentifier;
15 139 --          lastParent := NIL;
15 140 --
15 141 --          FOR i := 1 TO numViews DO
15 142 --              WITH theViewInfo^ DO
15 143 -- 2-              BEGIN
15 144 --                  {$IFC qDebug}
15 145 --                      IF gIntenseDebugging THEN
15 146 -- 3-                      BEGIN
15 147 --                          WrLbLSig('signature', itsSignature);
15 148 --                          WRITELN;
15 149 --                          WrLbLSig('itsParentID', itsParentID);
15 150 --                          WrLbLSig('thisViewID', thisViewID);
15 151 --                          WriteLn;
15 152 --                          WrLbLVpt('itsLocation', itsLocation);
15 153 --                          WrLbLVpt('itsSize', itsSize);
15 154 --                          Write('itsHSizeDet = ', ORD(itsHSizeDet):3);
15 155 --                          WriteLn('itsVSizeDet = ', ORD(itsVSizeDet):3);
15 156 --                          WrLbLBoolean('isEnabled', isEnabled);
15 157 --                          WriteLn;
15 158 --                          WriteLn('----- end of view -----');
15 159 -- 3-                      END;
15 160 --                  {$ENDC}
15 161 --
15 162 --                      IF LONGINT(itsParentID) = LONGINT(kNoIdentifier) THEN
15 163 --                          lastParent := parentView
15 164 --                      ELSE IF (LONGINT(itsParentID) <> LONGINT(lastParentID)) & (parentView <> NIL) THEN
15 165 -- 3-                          BEGIN
15 166 --                              lastParent := parentView.FindSubView(itsParentID);
15 167 --                          {$IFC qDebug}
15 168 --                              IF lastParent = NIL THEN
15 169 --                                  ProgramBreak('Unable to find parent view for template');
15 170 --                              {$ENDC}
15 171 -- 3-                          END;
15 172 --                          lastParentID := itsParentID;
15 173 --
15 174 --                          IF itsSignature = 'incl' THEN
15 175 -- 3-                          BEGIN
15 176 --                              aView := DoCreateViews(itsDocument, lastParent, includeRsrcID);
15 177 --                              OffsetPtr(theViewInfo, SIZEOF(ViewTemplate) - SIZEOF(Str255) + SIZEOF(INTEGER));
15 178 -- 3-                          END
15 179 --                          ELSE
15 180 --                              aView := CreateAView(itsDocument, lastParent, Ptr(theViewInfo));
15 181 --
15 182 --                          IF aView = NIL THEN
15 183 --                              LEAVE;
15 184 --                          IF parentView = NIL THEN
15 185 --                              parentView := aView;
15 186 --                          IF i = 1 THEN
15 187 --                              firstView := aView;
15 188 -- 2-                          END;
15 189 --
15 190 --                          HUnlock(Handle(aHandle));
15 191 --                          Success(fi);
15 192 --                          DoCreateViews := firstView;
15 193 -- 1-                          END;
15 194 -- 0 A      END;
15 195 --      {$ENDC}
15 196 --
15 197 --      {$S MRes}

```

```

15 198 -- {-----+
15 199 -- | DoCommandKey |
15 200 -- +-----+
15 201 -- A FUNCTION TEvtHandler.DoCommandKey (ch: Char; VAR info: EventInfo): TCommand;
15 202 -- A BEGIN
15 203 -- IF fNextHandler <> NIL THEN
15 204 -- DoCommandKey := fNextHandler.DoCommandKey(ch, info)
15 205 -- ELSE
15 206 -- DoCommandKey := gNoChanges;
15 207 -- A END;
15 208 --
15 209 -- {$$ MRes}
15 210 -- {-----+
15 211 -- | DoHandleEvent |
15 212 -- +-----+
15 213 -- A FUNCTION TEvtHandler.DoHandleEvent(nextEvent: PEventRecord;
15 214 -- VAR commandToPerform: TCommand): BOOLEAN;
15 215 -- A BEGIN
15 216 -- DoHandleEvent := FALSE;
15 217 -- A END;
15 218 --
15 219 -- {$IFC qTrace}{$D+}{$ENDC} (??? No longer want this compile-time stuff here ???)
15 220 -- {$$ MRes}
15 221 -- {-----+
15 222 -- | DoIdle |
15 223 -- +-----+
15 224 -- A PROCEDURE TEvtHandler.DoIdle(phase: IdlePhase);
15 225 -- A BEGIN
15 226 -- A END;
15 227 -- {$IFC qTrace}{$D+}{$ENDC}
15 228 --
15 229 -- {$$ MRes}
15 230 -- {-----+
15 231 -- | DoKeyCommand |
15 232 -- +-----+
15 233 -- A FUNCTION TEvtHandler.DoKeyCommand(ch: Char; aKeyCode: INTEGER; VAR info: EventInfo): TCommand;
15 234 -- A BEGIN
15 235 -- IF fNextHandler <> NIL THEN
15 236 -- DoKeyCommand := fNextHandler.DoKeyCommand(ch, aKeyCode, info)
15 237 -- ELSE
15 238 -- DoKeyCommand := gNoChanges; (??? give an error ???)
15 239 -- A END;
15 240 --
15 241 -- {$$ MRes}
15 242 -- {-----+
15 243 -- | DoMenuCommand |
15 244 -- +-----+
15 245 -- A FUNCTION TEvtHandler.DoMenuCommand(aCmdNumber: CmdNumber): TCommand;
15 246 -- A BEGIN
15 247 -- IF fNextHandler <> NIL THEN
15 248 -- DoMenuCommand := fNextHandler.DoMenuCommand(aCmdNumber)
15 249 -- ELSE
15 250 -- BEGIN
15 251 -- {$IFC qDebug}
15 252 -- WriteLn('No one handled the command ', aCmdNumber:1);
15 253 -- {$ENDC}
15 254 -- DoMenuCommand := gNoChanges;
15 255 -- END;
15 256 -- A END;
15 257 --
15 258 -- {$$ MRes}
15 259 -- {-----+
15 260 -- | DoMultiClick |
15 261 -- +-----+
15 262 -- A FUNCTION TEvtHandler.DoMultiClick(lastDownPt, newDownPt: Point): BOOLEAN;
15 263 -- A BEGIN
15 264 -- IF fNextHandler <> NIL THEN
15 265 -- DoMultiClick := fNextHandler.DoMultiClick(lastDownPt, newDownPt)
15 266 -- ELSE
15 267 -- DoMultiClick := ABS(lastDownPt.h - newDownPt.h) +
15 268 -- ABS(lastDownPt.v - newDownPt.v) <= 5; (??? arbitrary choice ???)
15 269 -- A END;
15 270 --
15 271 -- {$IFC qTrace}{$D+}{$ENDC}
15 272 -- {$$ MRes}
15 273 -- {-----+
15 274 -- | DoSetupMenus |
15 275 -- +-----+
15 276 -- A PROCEDURE TEvtHandler.DoSetupMenus;
15 277 -- A BEGIN
15 278 -- IF fNextHandler <> NIL THEN
15 279 -- fNextHandler.DoSetupMenus;
15 280 -- A END;
15 281 -- {$IFC qTrace}{$D+}{$ENDC}
15 282 --
15 283 -- {$$ MRes}
15 284 -- {-----+
15 285 -- | HandlesPrintingCommands |
15 286 -- +-----+
15 287 -- A FUNCTION TEvtHandler.HandlesPrintingCommands: BOOLEAN;
15 288 -- A BEGIN
15 289 -- IF fNextHandler <> NIL THEN
15 290 -- HandlesPrintingCommands := fNextHandler.HandlesPrintingCommands
15 291 -- ELSE
15 292 -- HandlesPrintingCommands := FALSE
15 293 -- A END;
15 294 --
15 295 -- {$IFC qDebug}
15 296 -- {$$ MRes}

```

```

15 297 --      ($IFC qTrace){$D+}($ENDC)
15 298 --      {-----+
15 299 --      |   IdentifySoftware   |
15 300 --      +-----+
15 301 -- A  PROCEDURE TEvtHandler.IdentifySoftware;
15 302 0- A  BEGIN
15 303 --      IF fNextHandler <> NIL THEN
15 304 --          fNextHandler.IdentifySoftware;
15 305 -0 A  END;
15 306 --      ($IFC qTrace){$D+}($ENDC)
15 307 --      ($ENDC)
15 308 --
15 309 --      {$S MAActivate}
15 310 --      {-----+
15 311 --      |   InstallSelection   |
15 312 --      +-----+
15 313 -- A  PROCEDURE TEvtHandler.InstallSelection(wasActive, beActive: BOOLEAN);
15 314 0- A  BEGIN
15 315 -0 A  END;
15 316 --
15 317 --      ($IFC qDebug)
15 318 --      {$S MAdDebug}
15 319 --      ($IFC qTrace){$D+}($ENDC)
15 320 --      {-----+
15 321 --      |   LookupSymbol       |
15 322 --      +-----+
15 323 -- A  FUNCTION TEvtHandler.LookupSymbol(VAR sym: String8): LONGINT;
15 324 0- A  BEGIN
15 325 --      IF fNextHandler <> NIL THEN
15 326 --          LookupSymbol := fNextHandler.LookupSymbol(sym)
15 327 --      ELSE
15 328 --          LookupSymbol := -1;
15 329 -0 A  END;
15 330 --      ($IFC qTrace){$D+}($ENDC)
15 331 --      ($ENDC)
15 332 --
15 333 --      {$S MATerminate}
15 334 --      {-----+
15 335 --      |   Terminate         |
15 336 --      +-----+
15 337 -- A  PROCEDURE TEvtHandler.Terminate;
15 338 0- A  BEGIN
15 339 -0 A  END;
13 3003 --      ($I UMacApp.TApplication.p)

```

```

16 1 --      {SP}
16 2 --      { UMacApp.TApplication.p }
16 3 --      { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
16 4 --
16 5 --      { $$ MAINit }
16 6 --      {-----+ This routine preserves the menu color table information
16 7 --      |   MAGetNewMBar      | which is rudely disposed of by GetNewMBar
16 8 --      +-----}
16 9 -- A FUNCTION MAGetNewMBar (menuRsrcID: INTEGER): Handle;
16 10 --
16 11 --      TYPE
16 12 --          MenuCRsrc = RECORD
16 13 --              numEntries:   INTEGER;
16 14 --              data:         ARRAY [1..1092] OF MCEnter;
16 15 --              END;
16 16 --          PMenuCRsrc = ^MenuCRsrc;
16 17 --          HMenuCRsrc = ^PMenuCRsrc;
16 18 --
16 19 --      VAR
16 20 --          theColorTab:      MCTableHandle;
16 21 --          theColorRsrc:      HMenuCRsrc;
16 22 --
16 23 0- A BEGIN
16 24 --      IF gConfiguration.hasColorQD THEN
16 25 --          theColorTab := GetMCInfo;
16 26 --          MAGetNewMBar := GetNewMBar(menuRsrcID);
16 27 --          IF gConfiguration.hasColorQD & (theColorTab <> NIL) THEN
16 28 1-              BEGIN
16 29 --                  HLock(Handle(theColorTab));
16 30 --                  SetMCEntries(GetHandleSize(Handle(theColorTab)) DIV SIZEOF(MCEnter), theColorTab^);
16 31 --                  HUnlock(Handle(theColorTab));
16 32 --                  DispMCInfo(theColorTab);
16 33 --              { $IFC FALSE }
16 34 --              { Now that the old and new worlds coincide, let's see if an 'mctb' with the same
16 35 --              number as the menu bar is around, and load it as a convenience to the programmer.
16 36 --              THIS ISN'T STANDARD ACCEPTED PRACTICE, but the worst that can happen is that an
16 37 --              errant 'mctb' resource for a single menu might be reloaded. }
16 38 --              theColorRsrc := HMenuCRsrc(GetResource('mctb', menuRsrcID));
16 39 --              IF theColorRsrc <> NIL THEN
16 40 2-                  BEGIN
16 41 --                      HLock(Handle(theColorRsrc));
16 42 --                      WITH theColorRsrc^^ DO
16 43 --                          SetMCEntries(numEntries, @data[1]);
16 44 --                      HUnlock(Handle(theColorRsrc));
16 45 --                      ReleaseResource(Handle(theColorRsrc));
16 46 2-                  END;
16 47 --              { $ENDC }
16 48 1-              END;
16 49 0- A END;
16 50 --
16 51 --      {*****}
16 52 --      { *      T A p p l i c a t i o n      * }
16 53 --      {*****}
16 54 --      { $$ MAINit }
16 55 --      {-----+
16 56 --      |   IApplication      |
16 57 --      +-----}
16 58 -- A PROCEDURE TApplication.IApplication(itsMainFileType: OSType);
16 59 --
16 60 --      TYPE
16 61 --          MenuBarHdl = ^MenuBarPtr;
16 62 --          MenuBarPtr = ^MenuBarRecord;
16 63 --          MenuBarRecord = RECORD
16 64 --              nMenus:      INTEGER;
16 65 --              menuID:      ARRAY [1..1000] OF INTEGER;
16 66 --              END;
16 67 --
16 68 --      VAR
16 69 --          menuID:      INTEGER;
16 70 --          aMenu:      MenuHandle;
16 71 --          mbar:      Handle;
16 72 --          hmbars:      MenuBarHdl;
16 73 --          i:      INTEGER;
16 74 --
16 75 0- A BEGIN
16 76 --          gApplication := SELF;
16 77 --          gAppDone := FALSE;
16 78 --          gSysWindowActive := FALSE;
16 79 --          gTarget := SELF;
16 80 --
16 81 --          IEvtHandler(NIL);
16 82 --
16 83 --          { $IFC qInspector }
16 84 --          MakeInspector;
16 85 --          AddObjectToInspector(SELF);
16 86 --          AddObjectToInspector(gNullPrintHandler);
16 87 --          AddObjectToInspector(gFreeWindowList);
16 88 --          { $ENDC }
16 89 --
16 90 --          gDocList := NewList;
16 91 --          { $IFC qDebug }
16 92 --          gDocList.SetEltType('TDocument');
16 93 --          { $ENDC }
16 94 --
16 95 --          gMainFileType := itsMainFileType;
16 96 --
16 97 --          gDocument := NIL;
16 98 --          gLastCommand := NIL;

```

```

16 99 --
16 100 --      gVarClipPicSize := FALSE;      { temporary }
16 101 --
16 102 --      IF NOT gFinderPrinting THEN
16 103 1-      BEGIN
16 104 --          mbar := MAGetNewMBar(gMBarDisplayed);
16 105 --          IF mbar <> NIL THEN
16 106 2-      BEGIN
16 107 --              SetMenuBar(mbar);
16 108 --              DisposHandle(mbar);
16 109 -2      END
16 110 --      ELSE
16 111 2-      BEGIN
16 112 --          { $IFC qDebug }
16 113 --          Writeln('The MBar ', gMBarDisplayed:1, ' resource was not specified. ');
16 114 --          ProgramBreak('You will not have any menus!');
16 115 --          { $ENDC }
16 116 -2      END;
16 117 --
16 118 --      { $IFC qDebug }
16 119 --          aMenu := GetMenu(mDebug);
16 120 --          IF aMenu <> NIL THEN
16 121 --              InsertMenu(aMenu, 0);
16 122 --      { $ENDC }
16 123 --
16 124 --          aMenu := GetResMenu(mApple);
16 125 --          IF aMenu <> NIL THEN
16 126 --              AddResMenu(aMenu, 'DRVr');
16 127 --
16 128 --          mbar := MAGetNewMBar(gMBarNotDisplayed);
16 129 --          DisposIfHandle(mbar);
16 130 --
16 131 --          IF gConfiguration.hasHierarchicalMenus THEN
16 132 2-      BEGIN
16 133 --          hmbar := MenuBarHdl(GetResource('MBar', gMBarHierarchical));
16 134 --          IF hmbar <> NIL THEN
16 135 3-      BEGIN
16 136 --              FOR i := 1 TO hmbar^.nMenus DO
16 137 4-      BEGIN
16 138 --                  aMenu := GetMenu(hmbar^.menuID[i]);
16 139 --                  IF aMenu <> NIL THEN
16 140 --                      InsertMenu(aMenu, -1);
16 141 -4      END;
16 142 --              DisposHandle(Handle(hmbar));
16 143 -3      END;
16 144 -2      END;
16 145 --
16 146 --          New(gClipOrphanage);
16 147 --          FailNil(gClipOrphanage);
16 148 --          gClipOrphanage.IDeskScrapView;
16 149 --
16 150 --          gClipWindow := MakeClipboardWindow;
16 151 -1      END;
16 152 --
16 153 --      { $IFC qDebug }
16 154 --          { These must occur after the application is initialized }
16 155 --          TRCGlobalHandle(@gTarget,      'gTarget');
16 156 --          TRCGlobalHandle(@gLastCommand,  'gLastCommand');
16 157 --          TRCGlobalHandle(@gDocument,     'gDocument');
16 158 --          TRCGlobalHandle(@gApplication,  'gApplication');
16 159 --          TRCGlobalHandle(@gDocList,     'gDocList');
16 160 --          TRCGlobalHandle(@gFreeWindowList, 'gFreeWindowList');
16 161 --          TRCGlobalHandle(@gClipView,    'gClipView');
16 162 --          TRCGlobalHandle(@gClipUndoView, 'gClipUndoView');
16 163 --          TRCGlobalHandle(@gPrintHandler, 'gPrintHandler');
16 164 --          TRCGlobalHandle(@gFrontWindow,  'gFrontWindow');
16 165 --          TRCGlobalHandle(@gFocusedView,  'gFocusedView');
16 166 --      { $ENDC }
16 167 -0 A  END;
16 168 --
16 169 --      { $S MAClipboard }
16 170 --      {-----+
16 171 --      | AbandonUndoClipboard |
16 172 --      +-----}
16 173 -- A  PROCEDURE TApplication.AbandonUndoClipboard;
16 174 0- A  BEGIN
16 175 --      IF gClipUndoView <> NIL THEN
16 176 1-      BEGIN
16 177 --      { $IFC qDebug }
16 178 --          IF gClipUndoView = gClipView THEN
16 179 --              ProgramBreak('About to Free view both in clip and undo Clip!');
16 180 --      { $ENDC }
16 181 --          gClipUndoView.FreeFromClipboard;
16 182 --          gClipUndoView := NIL;
16 183 -1      END;
16 184 -0 A  END;
16 185 --
16 186 --      { $S MAActivate }
16 187 --      {-----+
16 188 --      | AboutToLoseControl |
16 189 --      +-----}
16 190 -- A  PROCEDURE TApplication.AboutToLoseControl(convertClipboard: BOOLEAN);
16 191 --      LABEL 1000;
16 192 --      VAR
16 193 --          err: LONGINT;
16 194 --          fi:  FailInfo;
16 195 --
16 196 --      {-----+
16 197 --      | PublicizeFailed |

```





```

16 297 --                 IF docDirID = parmDirID THEN
16 298 --                 IF EqualString(fileName, doc.fTitle,
16 299 --                               ignoreCase, diacritSens) THEN
16 300 --                     result := doc;
16 301 --                 END;
16 302 --0 B      END;
16 303 --
16 304 --0 A      BEGIN
16 305 --         result := NIL;
16 306 --
16 307 --         parmVRefnum := volRefnum;
16 308 --         err := GetDirID(parmVRefnum, parmDirID);
16 309 --
16 310 --         IF err = noErr THEN
16 311 --             ForAllDocumentsDo(TestDoc);
16 312 --
16 313 --         AlreadyOpen := result;
16 314 --0 A      END;
16 315 --
16 316 --         {-----+
16 317 --         |   CallFileFilter
16 318 --         +-----+}
16 319 --0 A      FUNCTION CallFileFilter(paramBlock: HParamBlkPtr; routine: ProcPtr): BOOLEAN;
16 320 --         INLINE $205F, { MOVEA.L (A7)+,A0 }
16 321 --         $4E90, { JMP (A0) }
16 322 --
16 323 --         {$S MAFinder}
16 324 --         { This is called only when opening/printing from the finder; it simulates the
16 325 --         filtering done by Std File. }
16 326 --         {-----+
16 327 --         |   CanOpenDocument
16 328 --         +-----+}
16 329 --0 A      FUNCTION TApplication.CanOpenDocument(itsCmdNumber: CmdNumber; VAR anAppFile: AppFile): BOOLEAN;
16 330 --
16 331 --         VAR
16 332 --             dlgID:      INTEGER;
16 333 --             where:      Point;
16 334 --             fileFilter: ProcPtr;
16 335 --             dlgHook:    ProcPtr;
16 336 --             filterProc: ProcPtr;
16 337 --             typeList:   HTypeList;
16 338 --             i:          INTEGER;
16 339 --             paramBlock: HParamBlockRec;
16 340 --             numTypes:   INTEGER;
16 341 --0 A      BEGIN
16 342 --         CanOpenDocument := FALSE;
16 343 --
16 344 --         { First check that file type is in the list of allowed file types. See SFGetParms
16 345 --         below for more info. }
16 346 --         typeList := HTypeList(NewHandle(0));
16 347 --         FailNIL(typeList);
16 348 --
16 349 --         SFGetParms(itsCmdNumber, dlgID, where, fileFilter, dlgHook, filterProc, typeList);
16 350 --
16 351 --         numTypes := GetHandleSize(Handle(typeList)) DIV SIZEOF(ResType);
16 352 --         IF numTypes > 0 THEN { if 0 then want all types }
16 353 --             FOR i := 1 TO numTypes DO
16 354 --                 { do coercions because the compiler generates lousy code for comparing 2
16 355 --                 packed arrays of characters }
16 356 --                 IF LONGINT(anAppFile.fType) = LONGINT(typeList^[i]) THEN
16 357 --                     BEGIN
16 358 --                         IF fileFilter = NIL THEN
16 359 --                             CanOpenDocument := TRUE
16 360 --                         ELSE IF GetFileInfo(anAppFile.fName,
16 361 --                                           anAppFile.vRefnum, paramBlock) = noErr THEN
16 362 --                             CanOpenDocument := NOT CallFileFilter(@paramBlock, fileFilter)
16 363 --                         ELSE
16 364 --                             CanOpenDocument := FALSE;
16 365 --                         LEAVE;
16 366 --                     END;
16 367 --
16 368 --                 DisposHandle(Handle(typeList));
16 369 --0 A      END;
16 370 --
16 371 --         {$S MAREs}
16 372 --         {-----+
16 373 --         |   CheckDeskScrap
16 374 --         +-----+}
16 375 --0 A      PROCEDURE TApplication.CheckDeskScrap;
16 376 --         VAR
16 377 --             err:      OSErr;
16 378 --
16 379 --0 A      BEGIN
16 380 --         AbsorbScrapStuff;
16 381 --
16 382 --         IF (gOldScrapStuff.scrapCount <> gNewScrapStuff.scrapCount) THEN
16 383 --             BEGIN
16 384 --                 IF gLastCommand <> NIL THEN
16 385 --                     CommitLastCommand;
16 386 --                     gClipView.FreeFromClipboard; { AbandonCurrentClipboard }
16 387 --                     gClipView := NIL;           { no reason to have an Undo clipboard }
16 388 --
16 389 --                     { If the scrap is in memory and we are low on memory, then we
16 390 --                     dispose of the scrap (ZeroScrap) and use gClipOrphanage as
16 391 --                     the clipboard view. }
16 392 --                     IF (gNewScrapStuff.scrapState > 0) AND MemSpaceIsLow THEN
16 393 --                         BEGIN
16 394 --                             err := ZeroScrap; { ??? What should we do on an error? }
16 395 --                             AbsorbScrapStuff;

```

```

16 396 --          ShowError(memFullErr, msgImportClipFailed);
16 397 --          ClaimClipboard(gClipOrphanage);
16 398 --      END
16 399 --      ELSE
16 400 --          ReadFromDeskScrap;
16 401 --      END;
16 402 -- A  END;
16 403 --
16 404 --      {$S MAOpen}
16 405 --      {-----+
16 406 --      | ChooseDocument
16 407 --      +-----}
16 408 -- A  FUNCTION TApplication.ChooseDocument(itsCmdNumber: CmdNumber; VAR anAppFile: AppFile): BOOLEAN;
16 409 --      TYPE
16 410 --          HSFileTypeList = ^PSFileTypeList;
16 411 --          PSFileTypeList = ^SFileTypeList;
16 412 --
16 413 --      VAR
16 414 --          dlgID:      INTEGER;
16 415 --          where:      Point;
16 416 --          fileFilter: ProcPtr;
16 417 --          dlgHook:    ProcPtr;
16 418 --          filterProc: ProcPtr;
16 419 --          typeList:   HTypeList;
16 420 --          pTypeList:  PSFileTypeList;
16 421 --          numTypes:   INTEGER;
16 422 --          reply:      SFReply;
16 423 -- 0- A  BEGIN
16 424 --      typeList := HTypeList(NewHandle(0));
16 425 --      FailNIL(typeList);
16 426 --
16 427 --      SFGGetParms(itsCmdNumber, dlgID, where, fileFilter, dlgHook, filterProc, typeList);
16 428 --      numTypes := GetHandleSize(Handle(typeList)) DIV SIZEOF(ResType);
16 429 --
16 430 --      IF numTypes = 0 THEN
16 431 --      BEGIN
16 432 --          numTypes := -1;
16 433 --          pTypeList := @pTypeList; { arbitrary, as long as it points to 4 bytes of valid memory }
16 434 --      END
16 435 --      ELSE
16 436 --      BEGIN
16 437 --          MoveHHI(Handle(typeList)); { in case Std File does allocations }
16 438 --          HLock(Handle(typeList)); { ditto }
16 439 --          pTypeList := HSFileTypeList(typeList)^;
16 440 --      END;
16 441 --
16 442 --      {$IFC qDebug}
16 443 --      { Causes TApplication.GetEvent to call CheckRsrcUsage. }
16 444 --      gRsrcCheck := 0;
16 445 --      {$ENDC}
16 446 --
16 447 --      UpdateAllWindows; { needed to work around bug in SF; if all windows are
16 448 --                        not updated you wont be able to mount a disk
16 449 --                        correctly }
16 450 --      SFPGetFile(where, '', fileFilter, numTypes, pTypeList^, dlgHook, reply, dlgID, filterProc);
16 451 --
16 452 --      DisposalHandle(Handle(typeList));
16 453 --
16 454 --      ChooseDocument := reply.good;
16 455 --      IF reply.good THEN
16 456 --      BEGIN
16 457 --          anAppFile.vRefnum := reply.vRefnum;
16 458 --          anAppFile.fType := reply.fType;
16 459 --          anAppFile.versNum := reply.version;
16 460 --          anAppFile.fName := reply.fName;
16 461 --      END;
16 462 -- 0 A  END;
16 463 --
16 464 --      {$S MAClipboard}
16 465 --      {-----+
16 466 --      | ClaimClipboard
16 467 --      +-----}
16 468 -- A  PROCEDURE TApplication.ClaimClipboard(clipView: TView);
16 469 -- 0- A  BEGIN
16 470 --      AbandonUndoClipboard; { free up any old UNDO stuff }
16 471 --      gClipUndoView := gClipView; { Copy current clipboard contents to the Undo side }
16 472 --      IF clipView <> NIL THEN
16 473 --      BEGIN
16 474 --          {$IFC FALSE}
16 475 --          {$IFC qDebug}
16 476 --              WITH clipView DO
16 477 --      BEGIN
16 478 --          fShowBorders := TRUE; { ??? } { temp }
16 479 --          fShowExtraFeedback := TRUE; { ??? } { temp }
16 480 --      END;
16 481 --          {$ENDC}
16 482 --          {$ENDC}
16 483 --          SetClipView(clipView); { Will install it as gClipView }
16 484 --      END
16 485 --      ELSE
16 486 --      BEGIN
16 487 --          {$IFC qDebug}
16 488 --          ProgramBreak('Claiming clipboard with null view');
16 489 --          {$ENDC}
16 490 --      END;
16 491 --
16 492 --      gClipClaimed := TRUE;
16 493 -- 0 A  END;
16 494 --

```



```

16 594 --
16 595 --         gLastMsePt := where;
16 596 -1         END;
16 597 --
16 598 --         gLastClickPart := whereMouseDown;
16 599 --         gClickCount := clickCount;
16 600 --         CountClicks := clickCount;
16 601 -0 A     END;
16 602 --
16 603 --         {$$ MAClose}
16 604 --         {-----+
16 605 --         | DeleteDocument
16 606 --         +-----}
16 607 -- A     PROCEDURE TApplication.DeleteDocument(docToDelete: TDocument);
16 608 0- A     BEGIN
16 609 --         gDocList.Delete(docToDelete);
16 610 --         IF gDocument = docToDelete THEN
16 611 --             gDocument := NIL;
16 612 -0 A     END;
16 613 --
16 614 --         {$$ MAClose}
16 615 --         {-----+
16 616 --         | DeleteFreeWindow
16 617 --         +-----}
16 618 -- A     PROCEDURE TApplication.DeleteFreeWindow(windowToDelete: TWindow);
16 619 0- A     BEGIN
16 620 --         gFreeWindowList.Delete(windowToDelete);
16 621 -0 A     END;
16 622 --
16 623 --         {$$ MAREs}
16 624 --         {-----+
16 625 --         | DispatchEvent
16 626 --         +-----}
16 627 -- A     PROCEDURE TApplication.DispatchEvent(VAR theEventInfo: EventInfo; VAR commandToPerform: TCommand);
16 628 --
16 629 0- A     BEGIN
16 630 --         commandToPerform := gNoChanges;
16 631 --
16 632 --         WITH theEventInfo.thePEvt^ DO
16 633 1-         BEGIN
16 634 --
16 635 2-         CASE what OF
16 636 --             mouseUp:
16 637 --                 commandToPerform := HandleMouseUp(theEventInfo);
16 638 --
16 639 --             mouseDown:
16 640 --                 commandToPerform := HandleMouseDown(theEventInfo);
16 641 --
16 642 --             activateEvt:
16 643 --                 commandToPerform := HandleActivateEvent(theEventInfo);
16 644 --
16 645 --             updateEvt:
16 646 --                 commandToPerform := HandleUpdateEvent(theEventInfo);
16 647 --
16 648 --             keyDown, autoKey:
16 649 --                 commandToPerform := HandleKeyDownEvent(theEventInfo);
16 650 --
16 651 --             diskEvt:
16 652 --                 commandToPerform := HandleDiskEvent(theEventInfo);
16 653 --
16 654 --             app4Evt:
16 655 --                 { All app4Evt's are owned by the system }
16 656 --                 commandToPerform := HandleSystemEvent(theEventInfo);
16 657 --
16 658 --             OTHERWISE
16 659 --                 commandToPerform := HandleAlienEvent(theEventInfo);
16 660 --
16 661 -2         END; { case }
16 662 --
16 663 -1         END; { with }
16 664 -0 A     END;
16 665 --
16 666 --         {$$ MAREs}
16 667 --         {-----+
16 668 --         | DoCommandKey
16 669 --         +-----}
16 670 -- A     FUNCTION TApplication.DoCommandKey(ch: CHAR; VAR info: EventInfo): TCommand; OVERRIDE;
16 671 0- A     BEGIN
16 672 --         DoCommandKey := gNoChanges;
16 673 --         IF (NOT info.theAutoKey) & (NOT InModalMenuState) THEN
16 674 1-         BEGIN
16 675 --             SetupTheMenus;
16 676 --             DoCommandKey := MenuEvent(MenuKey(ch));
16 677 -1         END;
16 678 -0 A     END;
16 679 --
16 680 --         {$$ MANever}
16 681 --         {-----+
16 682 --         | DoMakeDocument
16 683 --         +-----}
16 684 -- A     FUNCTION TApplication.DoMakeDocument(itsCmdNumber: CmdNumber): TDocument;
16 685 --         (* VAR aYOURDocument: TDocument; *)
16 686 0- A     BEGIN
16 687 --         (*
16 688 --             New(aYOURDocument);
16 689 --             aYOURDocument.IYOURDocument(itsDocKind, ...);
16 690 --             DoMakeDocument := aYOURDocument;
16 691 --         *)
16 692 --         {$IFC qDebug}

```

```

16 693 --      ProgramBreak('You forgot to override TApplication.DoMakeDocument in your application');
16 694 --      {SENDC}
16 695 -- A      END;
16 696 --
16 697 --      {$S MASelCommand}
16 698 --      {-----+
16 699 --      |      DoMenuCommand
16 700 --      +-----}
16 701 -- A      FUNCTION TApplication.DoMenuCommand(aCmdNumber: CmdNumber): TCommand;
16 702 --      VAR
16 703 --          succeeded:      BOOLEAN;
16 704 --          aDocument:      TDocument;
16 705 --          deltaCount:      INTEGER;
16 706 --          aWindow:         TWindow;
16 707 --          anAppFile:       AppFile;
16 708 --          fi:              FailInfo;
16 709 --
16 710 --      {-----+
16 711 --      |      HdQuit
16 712 --      +-----}
16 713 -- B      PROCEDURE HdQuit(error: OSErr; message: LONGINT);
16 714 -- B      BEGIN
16 715 --          gAppDone := FALSE;
16 716 -- B      END;
16 717 --
16 718 -- A      BEGIN
16 719 --          DoMenuCommand := gNoChanges;
16 720 --
16 721 --      CASE aCmdNumber OF
16 722 --          cQuit:
16 723 --              BEGIN
16 724 --                  CatchFailures(fi, HdQuit);
16 725 --                  gAppDone := TRUE;
16 726 --                  Close;
16 727 --                  Success(fi);
16 728 --              END;
16 729 --
16 730 --          cNew..cNewLast:
16 731 --              OpenNew(aCmdNumber);
16 732 --
16 733 --          cOpen..cOpenLast:
16 734 --              BEGIN
16 735 --                  IF ChooseDocument(aCmdNumber, anAppFile) THEN
16 736 --                      OpenOld(aCmdNumber, anAppFile);
16 737 --              END;
16 738 --
16 739 --          cClose:
16 740 --              CloseWmgrWindow(FrontWindow);
16 741 --
16 742 --          cShowClipboard:
16 743 --              IF FrontWindow = gClipWindow.fWmgrWindow THEN
16 744 --                  gClipWindow.Close { Hide the clipboard }
16 745 --              ELSE
16 746 --              BEGIN
16 747 --                  gClipWindow.Open;
16 748 --                  gClipWindow.Select;
16 749 --              END;
16 750 --
16 751 --          cAboutApp:
16 752 --              StdAlert(phAboutApp);
16 753 --
16 754 --      {$IFC qWWNeeded}
16 755 --          cDebugWind:
16 756 --              BEGIN
16 757 --                  ShowWindow(gDebugWindowPtr);
16 758 --                  SelectWmgrWindow(gDebugWindowPtr);
16 759 --              END;
16 760 --      {$ENDC}
16 761 --
16 762 --      {$IFC qDebug}
16 763 --          cExperimenting:
16 764 --              gExperimenting := NOT gExperimenting;
16 765 --          cReportEvt:
16 766 --              gReportEvt := NOT gReportEvt;
16 767 --          cDebugPrinting:
16 768 --              gDebugPrinting := NOT gDebugPrinting;
16 769 --          cReportMenuChoices:
16 770 --              gReportMenuChoices := NOT gReportMenuChoices;
16 771 --          cIntenseDebugging:
16 772 --              gIntenseDebugging := NOT gIntenseDebugging;
16 773 --          cIdentifySoftware:
16 774 --              BEGIN
16 775 --                  WriteLn('----- Software Version(s): -----');
16 776 --                  gTarget.IdentifySoftware;
16 777 --              END;
16 778 --          cRefreshFrontWindow:
16 779 --              gFrontWindow.ForceRedraw;
16 780 --          cModalToggle:
16 781 --              gFrontWindow.fIsModal := NOT gFrontWindow.fIsModal;
16 782 --          cDoFirstClick:
16 783 --              gFrontWindow.fDoFirstClick := NOT gFrontWindow.fDoFirstClick;
16 784 --      {$ENDC}
16 785 --
16 786 --      {$IFC qTrace}
16 787 --          cTraceSetupMenus:
16 788 --              gTraceSetupMenus := NOT gTraceSetupMenus;
16 789 --          cTraceIdle:
16 790 --              gTraceIdle := NOT gTraceIdle;
16 791 --      {$ENDC}

```

```

16 792 --
16 793 --      {$IFC qInspector}
16 794 --          cNewInspectorWindow:
16 795 --              MakeInspectorWindow;
16 796 --      {$ENDC}
16 797 --
16 798 --          cUndo ( , cRedo ):
16 799 --              BEGIN
16 800 --                  IF gLastCommand.fChangesClipboard THEN
16 801 --                      SwapClipViews;
16 802 --
16 803 --                  IF gLastCommand.fCmdDone THEN
16 804 --                      BEGIN
16 805 --                          gLastCommand.UndoIt;
16 806 --                          deltaCount := -1;
16 807 --                      END
16 808 --                  ELSE
16 809 --                      BEGIN
16 810 --                          gLastCommand.RedoIt;
16 811 --                          deltaCount := 1;
16 812 --                      END;
16 813 --
16 814 --                      gLastCommand.fCmdDone := NOT gLastCommand.fCmdDone;
16 815 --
16 816 --                  IF gLastCommand.fCausesChange THEN { put this after .UndoIt/.RedoIt,
16 817 --                      so they can change the flag }
16 818 --                      WITH gLastCommand DO
16 819 --                          IF fChangedDocument <> NIL THEN
16 820 --                              WITH fChangedDocument DO
16 821 --                                  BEGIN
16 822 --                                      fChangeCount := fChangeCount + deltaCount;
16 823 --                                      {$IFC qDebug}
16 824 --                                      IF fChangeCount < 0 THEN
16 825 --                                          ProgramBreak('fChangeCount < 0');
16 826 --                                      {$ENDC}
16 827 --                                  END;
16 828 --                              END;
16 829 --                          END;
16 830 --                      OTHERWISE
16 831 --                          DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
16 832 --                      END; { case }
16 833 -- 0 A      END;
16 834 --
16 835 --      {$S MAREs}
16 836 --      {-----+
16 837 --      | DoSetupMenus
16 838 --      +-----}
16 839 -- 0 A      PROCEDURE TApplication.DoSetupMenus;
16 840 --      VAR
16 841 --          frontmost:      WindowPtr;
16 842 --
16 843 -- 0 A      BEGIN
16 844 --          INHERITED DoSetupMenus;
16 845 --
16 846 --          Enable(cAboutApp, TRUE);
16 847 --          Enable(cQuit, gEventLevel <= 1);      { Can't enable Quit if in nested event handling }
16 848 --          Enable(cShowClipboard, TRUE);
16 849 --          SetMenuState(cShowClipboard, kIDBuzzString, bzShowClip, bzHideClip,
16 850 --                      FrontWindow = gClipWindow.fWMgrWindow);
16 851 --
16 852 --          Enable(cNew, TRUE);
16 853 --          Enable(cOpen, TRUE);
16 854 --
16 855 --          {$IFC qWriteLnWin}
16 856 --              Enable(cDebugWind, gDebugWindowPtr <> NIL);
16 857 --          {$ENDC}
16 858 --
16 859 --          {$IFC qInspector}
16 860 --              Enable(cNewInspectorWindow, TRUE);
16 861 --          {$ENDC}
16 862 --
16 863 --          frontmost := FrontWindow;
16 864 --          IF frontmost <> NIL THEN
16 865 --              Enable(cClose, WindowPeek(frontmost).goAwayFlag <> FALSE);
16 866 -- 0 A      END;
16 867 --
16 868 --      {$S MAREs}
16 869 --      {-----+
16 870 --      | EachFreeWindow
16 871 --      +-----}
16 872 -- 0 A      PROCEDURE TApplication.EachFreeWindow(PROCEDURE DoToWindow(aWindow: TWindow));
16 873 -- 0 A      BEGIN
16 874 --          gFreeWindowList.Each(DoToWindow);
16 875 -- 0 A      END;
16 876 --
16 877 --      {$S MAREs}
16 878 --      {-----+
16 879 --      | EachHandler
16 880 --      +-----}
16 881 -- 0 A      PROCEDURE TApplication.EachHandler(aFirstHandler: TEvtHandler;
16 882 --          PROCEDURE DoToEvtHandler(anEvtHandler: TEvtHandler));
16 883 --          { Performs 'DoToEvtHandler' to aFirstHandler, then to its fNextHandler, etc.,
16 884 --            onward until the fNextHandler chain ends at NIL }
16 885 --
16 886 --      VAR
16 887 --          currHandler:      TEvtHandler;
16 888 --          nextHandler:      TEvtHandler;
16 889 --
16 890 -- 0 A      BEGIN

```

```

16 891 --      currHandler := aFirstHandler;
16 892 --      WHILE currHandler <> NIL DO
16 893 --          BEGIN
16 894 --              { Get next handler now, in case DoToEvtHandler frees currHandler }
16 895 --              nextHandler := currHandler.fNextHandler;
16 896 --              DoToEvtHandler(currHandler);
16 897 --              currHandler := nextHandler;
16 898 --          END;
16 899 --  A  END;
16 900 --
16 901 --      {IFC qDebug}
16 902 --      {$S MAFields}
16 903 --      {-----+
16 904 --      | Fields                                     |
16 905 --      +-----}
16 906 --  A  PROCEDURE TApplication.Fields (PROCEDURE DoToField (fieldName: Str255;
16 907 --                                          fieldAddr: Ptr;
16 908 --                                          fieldType: INTEGER)); OVERRIDE;
16 909 --
16 910 --  A  BEGIN
16 911 --      DoToField('TApplication', NIL, bClass);
16 912 --      DoToField('gAppDone', @gAppDone, bBoolean);
16 913 --      DoToField('gApplication', @gApplication, bObject);
16 914 --      DoToField('gBusyTempRgn', @gBusyTempRgn, bBoolean);
16 915 --      DoToField('gChooserOK', @gChooserOK, bBoolean);
16 916 --      DoToField('gClickCount', @gClickCount, bInteger);
16 917 --      DoToField('gClipClaimed', @gClipClaimed, bBoolean);
16 918 --      DoToField('gClipOrphanage', @gClipOrphanage, bObject);
16 919 --      DoToField('gClipUndoView', @gClipUndoView, bObject);
16 920 --      DoToField('gClipView', @gClipView, bObject);
16 921 --      DoToField('gClipWindow', @gClipWindow, bObject);
16 922 --      DoToField('gClipWrittenToDeskScrap', @gClipWrittenToDeskScrap, bBoolean);
16 923 --      DoToField('gCouldPrint', @gCouldPrint, bBoolean);
16 924 --      DoToField('gCurrPrintHandler', @gCurrPrintHandler, bObject);
16 925 --      DoToField('gCursorRgn', @gCursorRgn, bRgnHandle);
16 926 --      DoToField('gDebugPrinting', @gDebugPrinting, bBoolean);
16 927 --      DoToField('gDocument', @gDocument, bObject);
16 928 --      DoToField('gDocList', @gDocList, bObject);
16 929 --      DoToField('gDrawingPictScrap', @gDrawingPictScrap, bBoolean);
16 930 --      DoToField('gErrorParm3', @gErrorParm3, bString);
16 931 --      DoToField('gEventLevel', @gEventLevel, bInteger);
16 932 --      DoToField('gExperimenting', @gExperimenting, bBoolean);
16 933 --      DoToField('gFileCount', @gFileCount, bInteger);
16 934 --      DoToField('gFinderPrinting', @gFinderPrinting, bBoolean);
16 935 --      DoToField('gFocusedView', @gFocusedView, bObject);
16 936 --      DoToField('gFreeWindowList', @gFreeWindowList, bObject);
16 937 --      DoToField('gFrontWindow', @gFrontWindow, bObject);
16 938 --      DoToField('gGotClipType', @gGotClipType, bBoolean);
16 939 --      DoToField('gHeadCohandler', @gHeadCohandler, bObject);
16 940 --      DoToField('gIdleFreq', @gIdleFreq, bLongInt);
16 941 --      DoToField('gIdlePhase', @gIdlePhase, bByte);
16 942 --      DoToField('gInBackground', @gInBackground, bBoolean);
16 943 --      DoToField('gInitialized', @gInitialized, bBoolean);
16 944 --      DoToField('gIntenseDebugging', @gIntenseDebugging, bBoolean);
16 945 --      DoToField('gLastClickPart', @gLastClickPart, bInteger);
16 946 --      DoToField('gLastCommand', @gLastCommand, bObject);
16 947 --      DoToField('gLastDeskAcc', @gLastDeskAcc, bLongInt);
16 948 --      DoToField('gLastIdle', @gLastIdle, bLongInt);
16 949 --      DoToField('gLastMsePt', @gLastMsePt, bPoint);
16 950 --      DoToField('gLastUpTime', @gLastUpTime, bLongInt);
16 951 --      DoToField('gLongOffset', @gLongOffset, bVPoint);
16 952 --      DoToField('gLowSpaceInterval', @gLowSpaceInterval, bLongInt);
16 953 --      DoToField('gMainEventMask', @gMainEventMask, bHexInteger);
16 954 --      DoToField('gMainFileType', @gMainFileType, bOSType);
16 955 --      DoToField('gMBarDisplayed', @gMBarDisplayed, bInteger);
16 956 --      DoToField('gMBarHeight', @gMBarHeight, bInteger);
16 957 --      DoToField('gMBarHierarchical', @gMBarHierarchical, bInteger);
16 958 --      DoToField('gMBarNotDisplayed', @gMBarNotDisplayed, bInteger);
16 959 --      DoToField('gMenusAreSetup', @gMenusAreSetup, bBoolean);
16 960 --      DoToField('gNextSpaceMsg', @gNextSpaceMsg, bLongInt);
16 961 --      DoToField('gNoChanges', @gNoChanges, bObject);
16 962 --      DoToField('gNullPrintHandler', @gNullPrintHandler, bObject);
16 963 --      DoToField('gNumUntitled', @gNumUntitled, bInteger);
16 964 --      DoToField('gOldChooserFlag', @gOldChooserFlag, bBoolean);
16 965 --      DoToField('gOrthogonal[h]', @gOrthogonal[h], bByte);
16 966 --      DoToField('gOrthogonal[v]', @gOrthogonal[v], bByte);
16 967 --      DoToField('gPageOffset', @gPageOffset, bVPoint);
16 968 --      DoToField('gPrefClipType', @gPrefClipType, bOSType);
16 969 --      DoToField('gPrintHandler', @gPrintHandler, bObject);
16 970 --      DoToField('gPrinting', @gPrinting, bBoolean);
16 971 --      {IFC qTemplateViews}
16 972 --      DoToField('gProtoList', @gProtoList, bObject);
16 973 --      {SENDC}
16 974 --      DoToField('gRedrawMenuBar', @gRedrawMenuBar, bBoolean);
16 975 --      DoToField('gReportEvt', @gReportEvt, bBoolean);
16 976 --      DoToField('gReportMenuChoices', @gReportMenuChoices, bBoolean);
16 977 --      DoToField('gRsrcCheck', @gRsrcCheck, bBoolean);
16 978 --      DoToField('gSaveClip', @gSaveClip, bHandle);
16 979 --      DoToField('gSaveFocusedView', @gSaveFocusedView, bObject);
16 980 --      DoToField('gSaveLongOffset', @gSaveLongOffset, bVPoint);
16 981 --      DoToField('gSaveOrg', @gSaveOrg, bPoint);
16 982 --      DoToField('gSavePort', @gSavePort, bWindowPtr);
16 983 --      {IFC qTemplateViews}
16 984 --      DoToField('gSignatureCount', @gSignatureCount, bInteger);
16 985 --      {SENDC}
16 986 --      DoToField('gStdHysteresis', @gStdHysteresis, bPoint);
16 987 --      DoToField('gStdStaggerCount', @gStdStaggerCount, bInteger);
16 988 --      DoToField('gStdWMoveBounds', @gStdWMoveBounds, bRect);
16 989 --      DoToField('gStdWSizeRect', @gStdWSizeRect, bRect);

```



```

16 990 --      DoToField('gStdWScreenRect', @gStdWScreenRect, bRect);
16 991 --      DoToField('gSysWindowActive', @gSysWindowActive, bBoolean);
16 992 --      DoToField('gSystemStyle', @gSystemStyle, bTitle);
16 993 --      DoToField(' tsFont', @gSystemStyle.tsFont, bInteger);
16 994 --      DoToField(' tsFace', @gSystemStyle.tsFace, bStyle);
16 995 --      DoToField(' tsSize', @gSystemStyle.tsSize, bInteger);
16 996 --      DoToField(' tsColor', @gSystemStyle.tsColor, bRGBcolor);
16 997 --      DoToField('gTarget', @gTarget, bObject);
16 998 --      DoToField('gTempRgn', @gTempRgn, bHandle);
16 999 --      {$IFC qTrace}
16 1000 --      DoToField('gTraceIdle', @gTraceIdle, bBoolean);
16 1001 --      {$ENDC}
16 1002 --      DoToField('gUndoState', @gUndoState, bBoolean);
16 1003 --      DoToField('gUndoCmd', @gUndoCmd, bInteger);
16 1004 --      DoToField('gUsedBy', @gUsedBy, bString);
16 1005 --      DoToField('gVarClipPicSize', @gVarClipPicSize, bBoolean);
16 1006 --      DoToField('gWorkPort', @gWorkPort, bGrafPtr);
16 1007 --      DoToField('gWResSignature', @gWResSignature, bOSType);
16 1008 --      DoToField('gWResType', @gWResType, bString);
16 1009 --      DoToField('gZeroRect', @gZeroRect, bRect);
16 1010 --      DoToField('gZeroVPt', @gZeroVPt, bVRect);
16 1011 --      DoToField('gZeroVRect', @gZeroVRect, bVRect);
16 1012 --
16 1013 --      INHERITED Fields (DoToField);
16 1014 -0 A      END ( TApplication.Fields );
16 1015 --      {$ENDC}
16 1016 --
16 1017 --      {$S Mares}
16 1018 --      {-----+
16 1019 --      | FirstHandlerThat
16 1020 --      +-----}
16 1021 -- A      FUNCTION TApplication.FirstHandlerThat (aFirstHandler: TEvtHandler;
16 1022 --      FUNCTION TestEvtHandler (anEvtHandler: TEvtHandler): BOOLEAN): TEvtHandler;
16 1023 --      { Calls TestEvtHandler for each event handler until TestEvtHandler returns
16 1024 --      true. Returns the TEvtHandler object for which TestEvtHandler returned
16 1025 --      true, or NIL if TestEvtHandler never returned true. }
16 1026 --
16 1027 --      VAR
16 1028 --      currHandler: TEvtHandler;
16 1029 --      nextHandler: TEvtHandler;
16 1030 --
16 1031 0- A      BEGIN
16 1032 --      currHandler := aFirstHandler;
16 1033 --      WHILE currHandler <> NIL DO
16 1034 1-      BEGIN
16 1035 --      { Get next handler now, in case DoToEvtHandler free currHandler }
16 1036 --      nextHandler := currHandler.fNextHandler;
16 1037 --      IF TestEvtHandler(currHandler) THEN
16 1038 2-      BEGIN
16 1039 --      FirstHandlerThat := currHandler;
16 1040 --      EXIT (FirstHandlerThat);
16 1041 -2      END;
16 1042 --      currHandler := nextHandler;
16 1043 -1      END;
16 1044 --      FirstHandlerThat := NIL;
16 1045 -0 A      END;
16 1046 --
16 1047 --      {$S Mares}
16 1048 --      {-----+
16 1049 --      | ForAllDocumentsDo
16 1050 --      +-----}
16 1051 -- A      PROCEDURE TApplication.ForAllDocumentsDo (PROCEDURE DoToDoc (aDocument: TDocument));
16 1052 0- A      BEGIN
16 1053 --      gDocList.Each (DoToDoc);
16 1054 -0 A      END;
16 1055 --
16 1056 --      {$S Mares}
16 1057 --      {-----+
16 1058 --      | ForAllWindowsDo
16 1059 --      +-----}
16 1060 -- A      PROCEDURE TApplication.ForAllWindowsDo (PROCEDURE DoToWind (aWindow: TWindow));
16 1061 --      {-----+
16 1062 --      | DoToYourWindows
16 1063 --      +-----}
16 1064 --      PROCEDURE DoToYourWindows (aDocument: TDocument);
16 1065 0- B      BEGIN
16 1066 --      aDocument.ForAllWindowsDo (DoToWind);
16 1067 -0 B      END;
16 1068 0- A      BEGIN
16 1069 --      ForAllDocumentsDo (DoToYourWindows);
16 1070 --      EachFreeWindow (DoToWind);
16 1071 -0 A      END;
16 1072 --
16 1073 --      {$IFC qInspector}
16 1074 --      {$S MInspector}
16 1075 --      {-----+
16 1076 --      | GetInspectorName
16 1077 --      +-----}
16 1078 -- A      PROCEDURE TApplication.GetInspectorName (VAR inspectorName: Str255);
16 1079 --
16 1080 0- A      BEGIN
16 1081 --      IF SELF = gApplication THEN
16 1082 --      inspectorName := 'gApplication';
16 1083 -0 A      END;
16 1084 --      {$ENDC}
16 1085 --
16 1086 --      {$S MAClipboard}
16 1087 --      {-----+
16 1088 --      | GetDataToPaste

```

```

16 1089 -- +-----}
16 1090 -- A FUNCTION TApplication.GetDataToPaste(aDataHandle: Handle; VAR dataType: ResType): LONGINT;
16 1091 -- VAR
16 1092 --     err: LONGINT;
16 1093 --     myType: ResType;
16 1094 --
16 1095 0- A BEGIN
16 1096 --     IF gGotClipType THEN
16 1097 1-         BEGIN
16 1098 --             dataType := gPrefClipType;
16 1099 --
16 1100 --             err := gClipView.GivePasteData(aDataHandle, dataType);
16 1101 --
16 1102 --             IF err < 0 THEN
16 1103 --                 Failure(err, 0);
16 1104 -1             END
16 1105 --             ELSE
16 1106 1-                 BEGIN
16 1107 --                     ($IFC qDebug)
16 1108 --                     ProgramBreak('GetDataToPaste called when gGotClipType was FALSE');
16 1109 --                     ($ENDC)
16 1110 -1                 END;
16 1111 --
16 1112 --             GetDataToPaste := err;
16 1113 0- A END;
16 1114 --
16 1115 --     ($S MAREs)
16 1116 --     {-----}
16 1117 --     | GetEvent
16 1118 --     {-----}
16 1119 -- A FUNCTION TApplication.GetEvent(eventMask: INTEGER; sleep: LONGINT; cursorRgn: RgnHandle;
16 1120 --     VAR anEvent: EventRecord): BOOLEAN;
16 1121 --
16 1122 -- VAR
16 1123 --     haveEvent: BOOLEAN;
16 1124 --     dummyEvent: EventRecord;
16 1125 --
16 1126 0- A BEGIN
16 1127 --
16 1128 --     ($IFC qDebug)
16 1129 --     gRsrcCheck := gRsrcCheck - 1;
16 1130 --     IF gRsrcCheck <= 0 THEN
16 1131 1-         BEGIN
16 1132 --             CheckRsrcUsage;
16 1133 --             gRsrcCheck := kRsrcCheckInterval;
16 1134 -1         END;
16 1135 --     ($ENDC qDebug)
16 1136 --
16 1137 --     IF gConfiguration.hasWaitNextEvent THEN
16 1138 1-         BEGIN
16 1139 --         ($IFC qDebug)
16 1140 --             IF gIntenseDebugging & gReportEvt THEN
16 1141 2-                 BEGIN
16 1142 --                     WRITE('WaitNextEvent: sleep=', sleep:0);
16 1143 --                     IF cursorRgn = NIL THEN
16 1144 --                         WRITE(', cursor region=NIL')
16 1145 --                     ELSE
16 1146 --                         WrLblRect(', cursor', cursorRgn^^.rgnBBox);
16 1147 --                     WRITELN;
16 1148 -2                 END;
16 1149 --             ($ENDC)
16 1150 --             BusyActivate(FALSE); { Turn off busy cursor while we're away. }
16 1151 --
16 1152 --             IF sleep > 0 THEN
16 1153 --                 IF NOT EventAvail(eventMask, anEvent) AND { If no events pending, we may not come }
16 1154 --                     (gIdlePhase = idleBegin) THEN { _back for a while, so idle to make }
16 1155 2-                     BEGIN
16 1156 --                         Idle(idleBegin); { _sure menus, etc. reflect reality. }
16 1157 --                         gIdlePhase := idleContinue;
16 1158 -2                     END;
16 1159 --
16 1160 --             haveEvent := WaitNextEvent(eventMask, anEvent, sleep, cursorRgn);
16 1161 --
16 1162 --             { Here we decide what to do if we get a mouse-moved event. If there are NO pending
16 1163 --               events then we don't consider it an event and idle. If there ARE pending events,
16 1164 --               then we consider it an event in order to avoid idling. }
16 1165 --
16 1166 --             WITH anEvent DO
16 1167 --                 IF (what = app4Evt) & { Is this a mouse-moved event? }
16 1168 --                     (BAND(BSR(message, 24), $FF) = kMouseMovedMessage) &
16 1169 --                     (NOT EventAvail(eventMask, dummyEvent)) THEN
16 1170 2-                 BEGIN
16 1171 --                 ($IFC qDebug)
16 1172 --                     IF gReportEvt THEN
16 1173 --                         WriteLn('Mouse moved');
16 1174 --                 ($ENDC)
16 1175 --                     haveEvent := FALSE; { _but don't consider it a real event. }
16 1176 -2                     END;
16 1177 --
16 1178 --                 IF NOT gInBackground THEN { If we're not in the background, then }
16 1179 --                     BusyActivate(TRUE); { _enable the busy cursor mechanism. }
16 1180 -1                 END
16 1181 --             ELSE
16 1182 --
16 1183 --             BEGIN
16 1184 1-                 SystemTask;
16 1185 --                 haveEvent := GetNextEvent(eventMask, anEvent);
16 1186 --             ($IFC FALSE) { this is code for measuring event loop performance }
16 1187 --

```

```

16 1188 --      IF anEvent.what <> keyDown THEN
16 1189 --          IF gIdlePhase <> idleBegin THEN
16 1190 --              BEGIN
16 1191 --                  haveEvent := TRUE;
16 1192 --                  anEvent.what := keyDown;
16 1193 --                  anEvent.modifiers := 0;
16 1194 --              END
16 1195 --          ELSE
16 1196 --              haveEvent := FALSE;
16 1197 --      {SEND}
16 1198 --      END;
16 1199 --
16 1200 --      GetEvent := haveEvent;
16 1201 --
16 1202 -- A  END ( TApplication.GetEvent );
16 1203 --
16 1204 --      {$$ MOpen}
16 1205 --      ( ??? We should not muck with the window template; the extra code isn't worth it
16 1206 --        since programmer can easily change the resource file ??? )
16 1207 --      {-----+
16 1208 --      |  GetRsrcWindow
16 1209 --      +-----}
16 1210 -- A  FUNCTION TApplication.GetRsrcWindow(storage: Ptr; rsrcId: INTEGER;
16 1211 --                                     VAR isResizable, isClosable: BOOLEAN): WindowPtr;
16 1212 --      ( We force INVISIBLE in the WIND definition so the screen won't flash. )
16 1213 --
16 1214 --      TYPE
16 1215 --          WindowTemplate = RECORD
16 1216 --              bounds: RECT;
16 1217 --              procID: INTEGER;
16 1218 --              visible, filler1: BOOLEAN;
16 1219 --              goAway, filler2: BOOLEAN;
16 1220 --              refcon: LONGINT;
16 1221 --              itemsID: INTEGER; ( only for DLOG resource )
16 1222 --          END;
16 1223 --      PTemplate = ^WindowTemplate;
16 1224 --      HTemplate = ^PTemplate;
16 1225 --
16 1226 --      VAR
16 1227 --          aWmgrWindow: WindowPtr;
16 1228 --          templateHandle: HTemplate;
16 1229 --          rsrcType: ResType;
16 1230 --          dit1: Handle;
16 1231 --          oldPerm: BOOLEAN;
16 1232 --          fi: FailInfo;
16 1233 --
16 1234 --      {-----+
16 1235 --      |  Hd1Failure
16 1236 --      +-----}
16 1237 -- B  PROCEDURE Hd1Failure(error: INTEGER; message: LONGINT); ( ??? ERROR ??? )
16 1238 -- B  BEGIN
16 1239 --      ( Make sure the perm allocation flag is set back to what it was
16 1240 --        when we entered GetRsrcWindow. )
16 1241 --      oldPerm := PermAllocation(oldPerm);
16 1242 -- B  END;
16 1243 --
16 1244 --
16 1245 -- A  BEGIN
16 1246 --      oldPerm := PermAllocation(FALSE);
16 1247 --      ( Even though the window is permanent, we allocate it
16 1248 --        under a temporary flag so that the maximum memory
16 1249 --        is available. Quickdraw can blow up if it can't
16 1250 --        allocate a grafPort. )
16 1251 --
16 1252 --      CatchFailures(fi, Hd1Failure);
16 1253 --
16 1254 --      templateHandle := HTemplate(GetResource('WIND', rsrcId));
16 1255 --      FailNIL(templateHandle);
16 1256 --      MoveHHI(Handle(templateHandle)); ( in case it is locked by the ROM )
16 1257 --
16 1258 --      WITH templateHandle^^ DO
16 1259 --          BEGIN
16 1260 --      {SIFC NOT qNeedsROM128K}
16 1261 --          ( ignore request for zoomDocProc if not 128K ROM, because
16 1262 --            the user might be running pre-3.0 System, which can't
16 1263 --            handle zoomDocProc )
16 1264 --          IF NOT gConfiguration.hasROM128K THEN
16 1265 --              procID := BAND(procID, $FFF7);
16 1266 --      {SEND}
16 1267 --
16 1268 --          visible := FALSE;
16 1269 --          isClosable := goAway;
16 1270 --          isResizable := (procID = documentProc) OR (procID = zoomDocProc);
16 1271 --          ( If your own defProc is resizable, too, then after
16 1272 --            the call on GetRsrcWindow, set isResizable TRUE )
16 1273 --          END;
16 1274 --
16 1275 --      IF gConfiguration.hasColorOD THEN
16 1276 --          aWmgrWindow := WindowPtr(GetNewCWindow(rsrcId, Pointer(storage), Pointer(-1)));
16 1277 --      ELSE
16 1278 --          aWmgrWindow := GetNewWindow(rsrcId, Pointer(storage), Pointer(-1));
16 1279 --
16 1280 --      SetWRefCon(aWmgrWindow, templateHandle^^.procID); ( Added for purposes of window template writing )
16 1281 --
16 1282 --      oldPerm := PermAllocation(oldPerm);
16 1283 --      Success(fi); ( Don't need the failure handler since we've set the
16 1284 --                  perm allocation flag back. )
16 1285 --
16 1286 --      FailNIL(aWmgrWindow);

```

```

16 1287 --
16 1288 --      { Now we must make sure that the code reserve is still intact.}
16 1289 --      IF NOT CheckReserve THEN
16 1290 1-      BEGIN
16 1291 --          FreeWmgrWindow(aWmgrWindow, storage = NIL);
16 1292 --
16 1293 --          Failure(memFullErr, 0);
16 1294 -1      END;
16 1295 --
16 1296 --      GetRsrcWindow := aWmgrWindow;
16 1297 -0 A  END;
16 1298 --
16 1299 --      {$S MRes}
16 1300 --      {-----+
16 1301 --      |   HandleActivateEvent   |
16 1302 --      +-----+
16 1303 -- A  FUNCTION TApplication.HandleActivateEvent (VAR theEventInfo: EventInfo): TCommand;
16 1304 --
16 1305 --      VAR
16 1306 --          aWindow:      TWindow;
16 1307 --
16 1308 0- A  BEGIN
16 1309 --          WITH theEventInfo, thePEvent^ DO
16 1310 --              ($IFC qWWNeeded)
16 1311 --                  IF WindowPtr(message) = gDebugWindowPtr THEN
16 1312 --                      WWActivateEvent(modifiers)
16 1313 --                  ELSE
16 1314 --                      ($ENDC)
16 1315 1-      BEGIN
16 1316 --          aWindow := WMgrToWindow(WindowPtr(message));
16 1317 --          IF aWindow <> NIL THEN
16 1318 --              aWindow.Activate(Odd(modifiers));
16 1319 -1      END;
16 1320 --
16 1321 --      HandleActivateEvent := gNoChanges;
16 1322 -0 A  END ( TApplication.HandleActivateEvent );
16 1323 --
16 1324 --      {$S MRes}
16 1325 --      {-----+
16 1326 --      |   HandleAlienEvent   |
16 1327 --      +-----+
16 1328 -- A  FUNCTION TApplication.HandleAlienEvent (VAR theEventInfo: EventInfo): TCommand;
16 1329 --
16 1330 --      VAR
16 1331 --          aCommand:      TCommand;
16 1332 --          anEvtHandler:  TEvtHandler;
16 1333 --
16 1334 --      {-----+
16 1335 --      |   TakeEvent   |
16 1336 --      +-----+
16 1337 -- B  FUNCTION TakeEvent(anEvtHandler: TEvtHandler): BOOLEAN;
16 1338 0- B  BEGIN
16 1339 --      TakeEvent := anEvtHandler.DoHandleEvent(theEventInfo.thePEvent, aCommand);
16 1340 -0 B  END;
16 1341 --
16 1342 0- A  BEGIN
16 1343 --      aCommand := gNoChanges;
16 1344 --      anEvtHandler := FirstHandlerThat(gHeadCohandler, TakeEvent);
16 1345 --      HandleAlienEvent := aCommand;
16 1346 -0 A  END;
16 1347 --
16 1348 --      {$S MADoCommand}
16 1349 --      {-----+
16 1350 --      |   HandleDiskEvent   |
16 1351 --      +-----+
16 1352 -- A  FUNCTION TApplication.HandleDiskEvent (VAR theEventInfo: EventInfo): TCommand;
16 1353 --
16 1354 --      CONST
16 1355 --          topLeft = $00500070;
16 1356 --
16 1357 --      VAR
16 1358 --          err:      INTEGER;
16 1359 --
16 1360 0- A  BEGIN
16 1361 --          WITH theEventInfo.thePEvent^ DO
16 1362 --              IF HiWrd(message) <> noErr THEN
16 1363 1-      BEGIN
16 1364 --          err := DIBadMount(Point(topLeft), message); { ??? do something with the error ??? }
16 1365 --          ($IFC qDebug)
16 1366 --              IF err <> noErr THEN
16 1367 --                  Writeln('error from DIBadMount is ', err:1);
16 1368 --              ($ENDC)
16 1369 -1      END;
16 1370 --
16 1371 --      HandleDiskEvent := gNoChanges;
16 1372 --
16 1373 -0 A  END;
16 1374 --
16 1375 --      {$S MRes}
16 1376 --      {-----+
16 1377 --      |   HandleEvent   |
16 1378 --      +-----+
16 1379 -- A  PROCEDURE TApplication.HandleEvent (VAR theEvent: EventRecord);
16 1380 --
16 1381 --      VAR
16 1382 --          fi:      FailInfo;
16 1383 --          commandToPerform: TCommand;
16 1384 --          theEventInfo:  EventInfo;
16 1385 --

```

```

16 1386 --      {-----+
16 1387 --      | HandleFailure |
16 1388 --      +-----+
16 1389 -- B      PROCEDURE HandleFailure(error: OSErr; message: LONGINT);
16 1390 -- B      BEGIN
16 1391 --          PostHandleEvent(theEventInfo);
16 1392 -- B      END;
16 1393 --
16 1394 -- A      BEGIN
16 1395 --          WITH theEventInfo, theEvent DO
16 1396 --              BEGIN
16 1397 --                  thePEvent := @theEvent;
16 1398 --                  theBtnState := BTst(modifiers, 7 { btnState });
16 1399 --                  theCmdKey := BTst(modifiers, 8 { cmdKey });
16 1400 --                  theShiftKey := BTst(modifiers, 9 { shiftKey });
16 1401 --                  theAlphaLock := BTst(modifiers, 10 { alphaLock });
16 1402 --                  theOptionKey := BTst(modifiers, 11 { optionKey });
16 1403 --                  theControlKey := BTst(modifiers, 12);
16 1404 --                  theAutoKey := what = autoKey;
16 1405 --                  theClickCount := gClickCount;
16 1406 --              END;
16 1407 --
16 1408 --          ($IFC qDebug)
16 1409 --          IF gReportEvt AND ((theEventInfo.thePEvent^.what = activateEvt) OR
16 1410 --              (FrontWindow <> gDebugWindowPtr)) THEN
16 1411 --              ReportEvent(theEventInfo);
16 1412 --          ($ENDC)
16 1413 --
16 1414 --          CatchFailures(fi, HandleFailure);
16 1415 --
16 1416 --          DispatchEvent(theEventInfo, commandToPerform);
16 1417 --          IF commandToPerform <> gNoChanges THEN
16 1418 --              PerformCommand(commandToPerform);
16 1419 --
16 1420 --          Success(fi);
16 1421 --
16 1422 --          PostHandleEvent(theEventInfo);
16 1423 --
16 1424 -- B      END ( TApplication.HandleEvent );
16 1425 --
16 1426 --      ($S MAFinder)
16 1427 --      {-----+
16 1428 --      | HandleFinderRequest |
16 1429 --      +-----+
16 1430 -- A      PROCEDURE TApplication.HandleFinderRequest;
16 1431 --      LABEL 1, 2;
16 1432 --
16 1433 --      VAR
16 1434 --          i:          INTEGER;
16 1435 --          anAppFile:  AppFile;
16 1436 --          doc:         TDocument;
16 1437 --          continuePrinting: BOOLEAN;
16 1438 --          cmd:         CmdNumber;
16 1439 --          fi:          FailInfo;
16 1440 --
16 1441 --          { ??? need better error messages here ??? }
16 1442 --          {-----+
16 1443 --          | HdIORequest |
16 1444 --          +-----+
16 1445 -- B      PROCEDURE HdIORequest(error: OSErr; message: LONGINT);
16 1446 -- B      BEGIN
16 1447 --          IF error <> noErr THEN
16 1448 --              BEGIN
16 1449 --                  IF message = 0 THEN
16 1450 --                      BEGIN
16 1451 --                          gErrorParm3 := anAppFile.fName;
16 1452 --                          IF cmd = cFinderPrint THEN
16 1453 --                              message := msgPrintFailed
16 1454 --                          ELSE
16 1455 --                              message := msgOpenFailed;
16 1456 --                      END;
16 1457 --                      ShowError(error, message);
16 1458 --                  END;
16 1459 --                  GOTO 1; { continue the loop }
16 1460 -- B      END;
16 1461 --
16 1462 --          {-----+
16 1463 --          | HdINRequest |
16 1464 --          +-----+
16 1465 -- B      PROCEDURE HdINRequest(error: OSErr; message: LONGINT);
16 1466 -- B      BEGIN
16 1467 --          IF error <> noErr THEN
16 1468 --              ShowError(error, message);
16 1469 --              GOTO 2; { exit the method }
16 1470 -- B      END;
16 1471 -- A      BEGIN
16 1472 --          ($IFC qDebug)
16 1473 --          IF gExperimenting THEN
16 1474 --              WriteLn('File count: ', gFileCount:1);
16 1475 --          ($ENDC)
16 1476 --
16 1477 --          continuePrinting := TRUE;
16 1478 --
16 1479 --          IF gFileCount = 0 THEN
16 1480 --              BEGIN
16 1481 --                  CatchFailures(fi, HdINRequest);
16 1482 --                  OpenNew(cFinderNew);
16 1483 --                  Success(fi);
16 1484 --              END

```

```

16 1485 --      ELSE { it's an OPEN or PRINT of 1 or more existing files }
16 1486 1-      BEGIN
16 1487 --      IF gFinderPrinting THEN
16 1488 --      cmd := cFinderPrint
16 1489 --      ELSE
16 1490 --      cmd := cFinderOpen;
16 1491 --
16 1492 --      FOR i := 1 TO gFileCount DO
16 1493 2-      BEGIN
16 1494 --      CatchFailures(fi, HdlORquest);
16 1495 --
16 1496 --      GetAppFiles(i, anAppFile);
16 1497 --
16 1498 --      IF CanOpenDocument(cmd, anAppFile) THEN
16 1499 3-      BEGIN
16 1500 --      ClrAppFiles(i);
16 1501 --
16 1502 --      IF gFinderPrinting THEN
16 1503 4-      BEGIN
16 1504 --      IF continuePrinting THEN
16 1505 --      continuePrinting := PrintDocument(anAppFile);
16 1506 4-      END
16 1507 --      ELSE
16 1508 --      OpenOld(cFinderOpen, anAppFile);
16 1509 3-      END
16 1510 --      ELSE
16 1511 --      Failure(errNotMyType, 0);
16 1512 --
16 1513 --      Success(fi);
16 1514 --      1: { continue the loop }
16 1515 2-      END;
16 1516 1-      END;
16 1517 --      2: { exit the method }
16 1518 0 A END;
16 1519 --
16 1520 --      {$S MAREs}
16 1521 --      {-----+
16 1522 --      | HandleKeyDownEvent
16 1523 --      +-----+}
16 1524 0 A FUNCTION TApplication.HandleKeyDownEvent (VAR theEventInfo: EventInfo): TCommand;
16 1525 --
16 1526 --      VAR
16 1527 --      ch:          CHAR;
16 1528 --      keycode:    INTEGER;
16 1529 --
16 1530 0 A BEGIN
16 1531 --      WITH theEventInfo, thePEvent^ DO
16 1532 1-      BEGIN
16 1533 --      ch := CHR(BAND(message, charCodeMask));
16 1534 --      keyCode := BSR(BAND(message, keyCodeMask), 8);
16 1535 --
16 1536 --      IF theCmdKey THEN
16 1537 --      HandleKeyDownEvent := gTarget.DoCommandKey(ch, theEventInfo)
16 1538 --      ELSE
16 1539 --      HandleKeyDownEvent := gTarget.DoKeyCommand(ch, keycode, theEventInfo);
16 1540 1-      END;
16 1541 0 A END { TApplication.HandleKeyDown };
16 1542 --
16 1543 --      {$S MAREs}
16 1544 --      {-----+
16 1545 --      | HandleMouseDown
16 1546 --      +-----+}
16 1547 0 A FUNCTION TApplication.HandleMouseDown(VAR theEventInfo: EventInfo): TCommand;
16 1548 --
16 1549 --      VAR
16 1550 --      doClick:      BOOLEAN;
16 1551 --      aWindow:      TWindow;
16 1552 --      aWmgrWindow:  WindowPtr;
16 1553 --      whereMouseDown: INTEGER;
16 1554 --      sysWindowAct:  BOOLEAN;
16 1555 --      aCommand:      TCommand;
16 1556 --      theMouse:      Point;
16 1557 --      theVMouse:     VPoint;
16 1558 --      hysteresis:    Point;
16 1559 --
16 1560 0 A BEGIN
16 1561 --      HandleMouseDown := gNoChanges;
16 1562 --
16 1563 --      WITH theEventInfo, thePEvent^ DO
16 1564 1-      BEGIN
16 1565 --      whereMouseDown := FindWindow(where, aWmgrWindow);
16 1566 --      theClickCount := CountClicks(thePEvent, whereMouseDown);
16 1567 --
16 1568 --      {$IFC qDebug}
16 1569 --      IF gReportEvt AND (FrontWindow <> gDebugWindowPtr) AND (aWmgrWindow <> gDebugWindowPtr) THEN
16 1570 --      ReportMouseDown(whereMouseDown);
16 1571 --      {$ENDC}
16 1572 --
16 1573 --      aWindow := WMgrToWindow(aWmgrWindow);
16 1574 --
16 1575 --      IF ((whereMouseDown = inMenuBar) & InModalMenuState) |
16 1576 --      ((whereMouseDown <> inMenuBar) & InModalState & (aWindow <> gFrontWindow)) THEN
16 1577 2-      BEGIN
16 1578 --      SysBeep(2);
16 1579 --      EXIT(HandleMouseDown);
16 1580 2-      END;
16 1581 --
16 1582 --      {$IFC qWWNeeded}
16 1583 --      IF aWmgrWindow = gDebugWindowPtr THEN

```

```

16 1584 2-      BEGIN
16 1585 --      IF gDebugWindowPtr <> FrontWindow THEN
16 1586 --          SelectWMgrWindow(gDebugWindowPtr);
16 1587 --      WMouseDown(whereMouseDown, where, modifiers);
16 1588 --      EXIT(HandleMouseDown);
16 1589 -2      END;
16 1590 --      ($ENDC)
16 1591 -1      END;
16 1592 --
16 1593 --      IF whereMouseDown <> inContent THEN
16 1594 --          SetCursor(arrow);
16 1595 --
16 1596 --      WITH theEventInfo, thePEvent^ DO
16 1597 1-          CASE whereMouseDown OF
16 1598 --              inMenuBar:
16 1599 2-                  BEGIN
16 1600 --                      SetupTheMenus; { gives application a chance to setup individual menu items }
16 1601 --                      HandleMouseDown := MenuEvent(MenuSelect(where));
16 1602 -2                  END;
16 1603 --
16 1604 --              inSysWindow:
16 1605 --                  SystemClick(thePEvent^, aWmgrWindow);
16 1606 --
16 1607 --              inContent:
16 1608 --                  IF aWindow.Focus THEN { if we can't focus, we're in trouble }
16 1609 2-                      BEGIN
16 1610 --                          theMouse := where;
16 1611 --                          GlobalToLocal(theMouse);
16 1612 --                          aWindow.QDToViewPt(theMouse, theVMouse);
16 1613 --                          hysteresis := gStdHysteresis;
16 1614 --                          IF aWindow.HandleMouseDown(theVMouse, theEventInfo,
16 1615 --                                                  hysteresis, aCommand) THEN
16 1616 --                              IF aCommand <> gNoChanges THEN
16 1617 --                                  HandleMouseDown := TrackMouse(where, hysteresis, aCommand);
16 1618 -2                          END;
16 1619 --
16 1620 --                      inDrag:
16 1621 --                          aWindow.MoveByUser(where);
16 1622 --
16 1623 --                      inGrow:
16 1624 --                          aWindow.ResizeByUser(where);
16 1625 --
16 1626 --                      inGoAway:
16 1627 --                          IF TrackGoAway(aWmgrWindow, where) THEN
16 1628 --                              CloseWmgrWindow(aWmgrWindow); { ??? }
16 1629 --
16 1630 --                      inZoomIn, inZoomOut:
16 1631 --                          IF TrackBox(aWmgrWindow, where, whereMouseDown) THEN
16 1632 --                              aWindow.Zoom(whereMouseDown);
16 1633 --
16 1634 --                      inDesk:
16 1635 --                          ;
16 1636 --
16 1637 -1      END; { CASE }
16 1638 -0 A      END;
16 1639 --
16 1640 --      {$$ MAREs}
16 1641 --      {-----+
16 1642 --      |   HandleMouseUp
16 1643 --      +-----+}
16 1644 -- A      FUNCTION TApplication.HandleMouseUp (VAR theEventInfo: EventInfo): TCommand;
16 1645 --
16 1646 0- A      BEGIN
16 1647 --          { Remember time of last mouse up, in order to detect double clicks }
16 1648 --          gLastUpTime := theEventInfo.thePEvent^.when;
16 1649 --          HandleMouseUp := gNoChanges;
16 1650 -0 A      END { TApplication.HandleMouseUp };
16 1651 --
16 1652 --      {$$ MAREs}
16 1653 --      {-----+
16 1654 --      |   HandleSystemEvent
16 1655 --      +-----+}
16 1656 -- A      FUNCTION TApplication.HandleSystemEvent (VAR theEventInfo: EventInfo): TCommand;
16 1657 --
16 1658 --      VAR
16 1659 --          e:                OSErr;
16 1660 --          switchingIn:      BOOLEAN;
16 1661 --          convertClipboard:  BOOLEAN;
16 1662 --          aWmgrWindow:      WindowPtr;
16 1663 --          aWindow:          TWindow;
16 1664 --          anEvent:          EventRecord;
16 1665 --          dontCare:          BOOLEAN;
16 1666 --
16 1667 0- A      BEGIN
16 1668 --          WITH theEventInfo.thePEvent^ DO
16 1669 1-              CASE BSR(BAND(message, $FF000000), 24) OF
16 1670 --                  kSuspendOrResume:
16 1671 2-                      BEGIN
16 1672 --                          switchingIn := Odd(message);
16 1673 --                          convertClipboard := BAND(message, $00000002) <> 0;
16 1674 --
16 1675 --                          IF switchingIn THEN
16 1676 --                              RegainControl(convertClipboard)
16 1677 --                          ELSE
16 1678 --                              AboutToLoseControl(convertClipboard);
16 1679 --
16 1680 --                          aWmgrWindow := FrontWindow;
16 1681 --                      ($IFC qWWNeeded)
16 1682 --                      IF aWmgrWindow = gDebugWindowPtr THEN

```

```

16 1683 --      WWActivateEvent(ORD(switchingIn))
16 1684 --      ELSE
16 1685 --      ($ENDC)
16 1686 --      IF aWMgrWindow <> NIL THEN
16 1687 3-      BEGIN
16 1688 --      aWindow := WMgrToWindow(aWMgrWindow);
16 1689 --      IF aWindow <> NIL THEN
16 1690 --      aWindow.Activate(switchingIn);
16 1691 -3      END;
16 1692 --      gInBackground := NOT switchingIn;
16 1693 -2      END;
16 1694 -1      END;
16 1695 --
16 1696 --      HandleSystemEvent := gNoChanges;
16 1697 -0 A  END;
16 1698 --
16 1699 --      {$S MARES}
16 1700 --      {-----+
16 1701 --      |   HandleUpdateEvent   |
16 1702 --      +-----+}
16 1703 -- A  FUNCTION TApplication.HandleUpdateEvent (VAR theEventInfo: EventInfo): TCommand;
16 1704 --
16 1705 --      VAR
16 1706 --      aWindow:      TWindow;
16 1707 --
16 1708 0- A  BEGIN
16 1709 --      WITH theEventInfo.thePEvent^ DO
16 1710 1-      BEGIN
16 1711 --      {$IFC qWWNeeded}
16 1712 --      IF WindowPtr(message) = gDebugWindowPtr THEN
16 1713 --      WWUpdateEvent
16 1714 --      ELSE
16 1715 --      ($ENDC)
16 1716 2-      BEGIN
16 1717 --      aWindow := WMgrToWindow(WindowPtr(message));
16 1718 --      IF aWindow <> NIL THEN
16 1719 --      aWindow.Update;
16 1720 -2      END;
16 1721 -1      END;
16 1722 --      HandleUpdateEvent := gNoChanges;
16 1723 -0 A  END ( TApplication.HandleUpdateEvent );
16 1724 --
16 1725 --      {$IFC qDebug}
16 1726 --      {$S MAdDebug}
16 1727 --      {$IFC qTrace}{$D+}{$ENDC}
16 1728 --      {-----+
16 1729 --      |   IdentifySoftware   |
16 1730 --      +-----+}
16 1731 -- A  PROCEDURE TApplication.IdentifySoftware;
16 1732 0- A  BEGIN
16 1733 --      Writeln('UMacApp of ', COMPDATE, ' @ ', COMPTIME);
16 1734 --      IDUObject;
16 1735 --      IDUWritelnWindow;
16 1736 --      IDUTrace;
16 1737 -0 A  END;
16 1738 --      {$IFC qTrace}{$D++}{$ENDC}
16 1739 --      {$ENDC}
16 1740 --
16 1741 --      {$S MARES}
16 1742 --      {-----+
16 1743 --      |   Idle   |
16 1744 --      +-----+}
16 1745 -- A  PROCEDURE TApplication.Idle(phase: IdlePhase);
16 1746 --
16 1747 --      VAR
16 1748 --      currTick:      LONGINT;
16 1749 --      dontCare:      BOOLEAN;
16 1750 --
16 1751 --      {-----+
16 1752 --      |   DoIdleAction   |
16 1753 --      +-----+}
16 1754 -- B  PROCEDURE DoIdleAction(anEvtHandler: TEventHandler);
16 1755 --
16 1756 0- B  BEGIN
16 1757 --      ( If this handler needs idling, and enough ticks have elapsed
16 1758 --      since the last time it was idled, call its DoIdle. (This was not
16 1759 --      made a TEventHandler method in order to optimize idling speed.) )
16 1760 --      WITH anEvtHandler DO
16 1761 --      IF fIdleFreq <> kMaxIdleTime THEN { Does it idle at all? }
16 1762 1-      BEGIN
16 1763 --      IF (phase = idleBegin) | (currTick >= fLastIdle + fIdleFreq) THEN
16 1764 2-      BEGIN
16 1765 --      anEvtHandler.DoIdle(phase);
16 1766 --      fLastIdle := currTick;
16 1767 -2      END;
16 1768 --      gIdleFreq := Min(fIdleFreq, gIdleFreq);
16 1769 -1      END;
16 1770 -0 B  END;
16 1771 --
16 1772 0- A  BEGIN
16 1773 --
16 1774 --      currTick := TickCount;
16 1775 --
16 1776 --      IF phase = idleBegin THEN
16 1777 1-      BEGIN
16 1778 --      {$IFC qTrace}
16 1779 --      TRCEnable(gTraceIdle); { Trace during idle only if user wants to. }
16 1780 --      {$ENDC}
16 1781 --      IF MemSpaceIsLow THEN

```



```

16 1782 --          SpaceIsLow
16 1783 --      ELSE
16 1784 --          gNextSpaceMsg := currTick;
16 1785 --
16 1786 --          SetupTheMenus;          { To get the menu bar redrawn if necessary. }
16 1787 --          gLastIdle := 0;         { Force idling event handlers & co-handlers. }
16 1788 --          gIdleFreq := 0;
16 1789 --      END;
16 1790 --
16 1791 --      IF currTick >= gLastIdle + gIdleFreq THEN
16 1792 --      BEGIN
16 1793 --          gIdleFreq := kMaxIdleTime;
16 1794 --          EachHandler(gHeadCohandler, DoIdleAction);
16 1795 --          EachHandler(gTarget, DoIdleAction);
16 1796 --          gLastIdle := currTick;
16 1797 --      END;
16 1798 --
16 1799 --      IF phase <> idleEnd THEN
16 1800 --          dontCare := TrackCursor
16 1801 --      ($IFC qTrace)
16 1802 --      ELSE
16 1803 --          TRCEnable(TRUE)          { Make sure tracing back on at end of idle. }
16 1804 --      ($ENDC)
16 1805 --      ;
16 1806 --
16 1807 -- 0 A END;
16 1808 --
16 1809 --      {-----+
16 1810 --      | InModalState |
16 1811 --      +-----+
16 1812 --      {$S MAREs}
16 1813 --  A FUNCTION TApplication.InModalState: BOOLEAN;
16 1814 --
16 1815 -- 0 A BEGIN
16 1816 --          InModalState := (gFrontWindow <> NIL) & (gFrontWindow.fIsModal);
16 1817 -- 0 A END { TApplication.InModalState };
16 1818 --
16 1819 --      {-----+
16 1820 --      | InModalMenuState |
16 1821 --      +-----+
16 1822 --      {$S MAREs}
16 1823 --  A FUNCTION TApplication.InModalMenuState: BOOLEAN;
16 1824 --
16 1825 -- 0 A BEGIN
16 1826 --          InModalMenuState := (gFrontWindow <> NIL) & (NOT gFrontWindow.AllowsMenuAccess);
16 1827 -- 0 A END { TApplication.InModalMenuState };
16 1828 --
16 1829 --
16 1830 --      {$S MANonRes}
16 1831 --      {-----+
16 1832 --      | InstallCohandler |
16 1833 --      +-----+
16 1834 --  A PROCEDURE TApplication.InstallCohandler(aCohandler: TEvtHandler; addIt: BOOLEAN);
16 1835 --      VAR
16 1836 --          prevCohandler: TEvtHandler;
16 1837 --          currCohandler: TEvtHandler;
16 1838 -- 0 A BEGIN
16 1839 --          IF addIt THEN
16 1840 --          BEGIN
16 1841 --              aCohandler.fNextHandler := gHeadCohandler;
16 1842 --              gHeadCohandler := aCohandler;
16 1843 --          END
16 1844 --          ELSE
16 1845 --          BEGIN
16 1846 --              prevCohandler := NIL;
16 1847 --              currCohandler := gHeadCohandler;
16 1848 --              WHILE currCohandler <> NIL DO
16 1849 --              BEGIN
16 1850 --                  IF currCohandler = aCohandler THEN { found it }
16 1851 --                  BEGIN
16 1852 --                      IF prevCohandler = NIL THEN
16 1853 --                          gHeadCohandler := aCohandler.fNextHandler
16 1854 --                      ELSE
16 1855 --                          prevCohandler.fNextHandler := aCohandler.fNextHandler;
16 1856 --                          currCohandler := NIL; { So loop will end }
16 1857 --                      END
16 1858 --                  ELSE
16 1859 --                  BEGIN
16 1860 --                      prevCohandler := currCohandler;
16 1861 --                      currCohandler := currCohandler.fNextHandler;
16 1862 --                  END;
16 1863 --              END;
16 1864 --          END;
16 1865 -- 0 A END;
16 1866 --
16 1867 --      {$S MAREs}
16 1868 --      {-----+
16 1869 --      | IsDeskAccessory |
16 1870 --      +-----+
16 1871 --  A FUNCTION TApplication.IsDeskAccessory (aWMgrWindow: WindowPtr): BOOLEAN;
16 1872 -- 0 A BEGIN
16 1873 --          IF aWMgrWindow <> NIL THEN
16 1874 --              IsDeskAccessory := WindowPeek(aWMgrWindow)^.windowKind < 0
16 1875 --          ELSE
16 1876 --              IsDeskAccessory := FALSE;
16 1877 -- 0 A END;
16 1878 --
16 1879 --      {$S MAOpen}
16 1880 --      {-----+

```

```

16 1881 -- | KindOfDocument |
16 1882 -- +-----+
16 1883 -- A FUNCTION TApplication.KindOfDocument(itsCmdNumber: CmdNumber; itsPAppFile: PAppFile): CmdNumber;
16 1884 -- A BEGIN
16 1885 --     KindOfDocument := itsCmdNumber;
16 1886 -- A END;
16 1887 --
16 1888 -- {$S MAINit}
16 1889 -- {-----+
16 1890 -- | LaunchClipboard |
16 1891 -- +-----+
16 1892 -- A PROCEDURE TApplication.LaunchClipboard;
16 1893 -- A BEGIN
16 1894 --     AbsorbScrapStuff; { Get current scrapCount as a baseline }
16 1895 --     ReadFromDeskScrap;
16 1896 -- A END;
16 1897 --
16 1898 -- {$S Main} { must be in the main segment }
16 1899 -- {-----+
16 1900 -- | MainEventLoop |
16 1901 -- +-----+
16 1902 -- A PROCEDURE TApplication.MainEventLoop;
16 1903 -- A BEGIN
16 1904 --     gIdlePhase := idleBegin;
16 1905 --     REPEAT
16 1906 --         IF gIdlePhase = idleBegin THEN
16 1907 --             UnloadAllSegments; { don't unload segments after idle has begun }
16 1908 --
16 1909 --             { ??? should we (1) unload segs after completing idle but before doing the event?
16 1910 --               (2) unload segs while processing event during background printing? }
16 1911 --
16 1912 --             PollEvent;
16 1913 --         UNTIL
16 1914 --             gAppDone; { gAppDone is a global BOOLEAN; that we set TRUE when the user chooses 'Quit' }
16 1915 -- A END;
16 1916 --
16 1917 -- {$S MAINit}
16 1918 -- {-----+
16 1919 -- | MakeClipboardWindow |
16 1920 -- +-----+
16 1921 -- A FUNCTION TApplication.MakeClipboardWindow: TWindow;
16 1922 --
16 1923 -- A BEGIN
16 1924 --     {$IFC qTemplateViews}
16 1925 --         MakeClipboardWindow := NewTemplateWindow(kIDClipWindow, NIL);
16 1926 --     {$ELSEC}
16 1927 --         MakeClipboardWindow := NewSimpleWindow(kIDClipWindow, TRUE, TRUE, NIL, NIL);
16 1928 --     {$ENDC}
16 1929 -- A END;
16 1930 --
16 1931 -- {$S MAClipboard}
16 1932 -- {-----+
16 1933 -- | MakeViewForAlienClipboard |
16 1934 -- +-----+
16 1935 -- A FUNCTION TApplication.MakeViewForAlienClipboard: TView;
16 1936 --
16 1937 -- A BEGIN
16 1938 --     { If the application doesn't override this then we just set the
16 1939 --       clipboard view to the orphanage, which handles TEXT and PICT
16 1940 --       scraps in a standard way. }
16 1941 --     MakeViewForAlienClipboard := gClipOrphanage;
16 1942 -- A END;
16 1943 --
16 1944 -- {$S MASelCommand}
16 1945 -- {-----+
16 1946 -- | MenuEvent |
16 1947 -- +-----+
16 1948 -- A FUNCTION TApplication.MenuEvent(menuItem: LONGINT): TCommand;
16 1949 --
16 1950 --     VAR
16 1951 --         fi:          FailInfo;
16 1952 --         alreadyDidCommand: BOOLEAN;
16 1953 --         newCommand:   TCommand;
16 1954 --         cmd:           CmdNumber;
16 1955 --         s:             Str255;
16 1956 --         theMenuNumber: INTEGER;
16 1957 --         theItemNumber: INTEGER;
16 1958 --
16 1959 --     {-----+
16 1960 --     | Hd1MenuEvt |
16 1961 --     +-----+
16 1962 -- B PROCEDURE Hd1MenuEvt(error: OSErr; message: LONGINT);
16 1963 -- B BEGIN
16 1964 --     IF gSysWindowActive THEN
16 1965 --         BusyActivate(FALSE);
16 1966 --
16 1967 --         FailNewMessage(error, message, BuildMessage(cmd, msgCmdErr));
16 1968 -- B END;
16 1969 --
16 1970 -- A BEGIN
16 1971 --     MenuEvent := gNoChanges;
16 1972 --     theMenuNumber := HiWrd(menuItem);
16 1973 --     theItemNumber := LoWrd(menuItem);
16 1974 --
16 1975 --     alreadyDidCommand := FALSE;
16 1976 --     IF theMenuNumber <> 0 THEN
16 1977 --         BEGIN
16 1978 --
16 1979 --             cmd := CmdFromMenuItem(theMenuNumber, theItemNumber);

```

```

16 1980 --      ($IFC qDebug)
16 1981 --          IF cmd = cCantUndo THEN
16 1982 2-              BEGIN
16 1983 --                  WriteLn('Command number ', cCantUndo:1,
16 1984 --                      ' is reserved for MacApp. ');
16 1985 --                  ProgramBreak('Use of reserved command number. ');
16 1986 -2              END;
16 1987 --
16 1988 --          IF gReportMenuChoices AND (cmd > 0) THEN
16 1989 --              WriteLn('Menu Choice Command Number = ', cmd:1);
16 1990 --      ($ENDC qDebug)
16 1991 --
16 1992 --          IF (cmd >= cEditBase) AND (cmd <= cEditLast) THEN
16 1993 --
16 1994 --              alreadyDidCommand := SystemEdit(cmd - cEditBase)
16 1995 --
16 1996 --          ELSE IF (cmd < 0) AND (theMenuNumber = mApple) THEN
16 1997 2-              BEGIN
16 1998 --                  GetItem(GetMHandle(mApple), theItemNumber, s);
16 1999 --                  OpenDeskAccessory(s);
16 2000 --                  alreadyDidCommand := TRUE;
16 2001 -2              END;
16 2002 --
16 2003 --          IF NOT alreadyDidCommand THEN
16 2004 2-              BEGIN
16 2005 --                  CatchFailures(fi, HdMenuEvt);
16 2006 --
16 2007 --                  IF gSysWindowActive THEN
16 2008 --                      BusyActivate(TRUE);
16 2009 --
16 2010 --                      MenuEvent := gTarget.DoMenuCommand(cmd);
16 2011 --
16 2012 --                  IF gSysWindowActive THEN
16 2013 --                      BusyActivate(FALSE);
16 2014 --
16 2015 --                      Success(fi);
16 2016 -2                  END;
16 2017 -1              END;
16 2018 -0 A      END;
16 2019 --
16 2020 --      ($$ MASElCommand)
16 2021 --      {-----+
16 2022 --      | OpenDeskAccessory
16 2023 --      +-----}
16 2024 -- A      PROCEDURE TApplication.OpenDeskAccessory (deskAccName: Str255);
16 2025 --
16 2026 --      VAR
16 2027 --          aRefNum:          INTEGER;
16 2028 --          drvvrH:           Handle;
16 2029 --          theID:            INTEGER;
16 2030 --          theType:          ResType;
16 2031 --          theName:          Str255;
16 2032 --          oldPerm:          BOOLEAN;
16 2033 --          ourHeap:          BOOLEAN;
16 2034 --          fi:               FailInfo;
16 2035 --          err:              OSErr;
16 2036 --
16 2037 -- B      PROCEDURE HdOpenDeskAcc(error: OSErr; message: LONGINT);
16 2038 0- B      BEGIN
16 2039 --          IF aRefNum <> 0 THEN
16 2040 --              CloseDeskAcc(aRefNum);
16 2041 --
16 2042 --          IF message = 0 THEN
16 2043 1-              BEGIN
16 2044 --                  gErrorParm3 := deskAccName;
16 2045 --                  { Get rid. of leading null character }
16 2046 --                  IF Ord(gErrorParm3[1]) = 0 THEN
16 2047 --                      Delete(gErrorParm3, 1, 1);
16 2048 -1                  END;
16 2049 --
16 2050 --                  FailNewMessage(error, message, msgOpenFailed);
16 2051 -0 B      END;
16 2052 --
16 2053 -- B      FUNCTION IsOpen (itsID: INTEGER): BOOLEAN;
16 2054 --
16 2055 --      CONST
16 2056 --          UnitNtryCnt =    $01D2;
16 2057 --          UTableBase =    $011C;
16 2058 --
16 2059 --      TYPE
16 2060 --          UnitTable =      ARRAY [0..128] OF DCtlHandle;
16 2061 --          UnitTablePtr =   ^UnitTable;
16 2062 --
16 2063 --      VAR
16 2064 --          dceHnd:          DCtlHandle;
16 2065 --
16 2066 0- B      BEGIN
16 2067 --          IsOpen := FALSE;
16 2068 --          IF (itsID >= 0) AND (itsID < PInteger(UnitNtryCnt)^) THEN
16 2069 1-              BEGIN
16 2070 --                  dceHnd := UnitTablePtr(PLongInt(UTableBase)^)[itsID];
16 2071 --                  IF (dceHnd <> NIL) & BTST(dceHnd^.dCtlFlags, 5) THEN
16 2072 --                      IsOpen := TRUE;
16 2073 -1                  END;
16 2074 -0 B      END;
16 2075 --
16 2076 0- A      BEGIN
16 2077 --          CatchFailures(fi, HdOpenDeskAcc);
16 2078 --          aRefNum := 0;

```

( Make sure failure handler works. )

```

16 2079 --
16 2080 --      { Attempt to load the DA into memory.  If 'deskAccName' refers to another app  }
16 2081 --      { rather than a read desk acc, then GetNamedResource returns a faked up handle }
16 2082 --      { courtesy of MultiFinder.  We open the DA with permanent allocation so as to  }
16 2083 --      { ensure that we don't take space from our code segments.                  }
16 2084 --
16 2085 --      oldPerm := PermAllocation(TRUE);
16 2086 --      drvvrH := GetNamedResource('DRVVR', deskAccName);
16 2087 --      oldPerm := PermAllocation(oldPerm);
16 2088 --
16 2089 --      IF drvvrH = NIL THEN                                { Either there wasn't enough memory }
16 2090 --          IF ResError <> noErr THEN                      { _to load the DA, or something is  }
16 2091 --              FailResError                               { _seriously wrong.                }
16 2092 --          ELSE
16 2093 --              Failure(resNotFound, 0);
16 2094 --
16 2095 --      { At this point if we are really opening a DA we know it fits in memory.      }
16 2096 --
16 2097 --      GetResInfo(drvvrH, theID, theType, theName); { If it's a not a real DA then this }
16 2098 --                                                  { will generate a ResError.      }
16 2099 --      ourHeap := HandleZone(drvvrH) = ApplicZone; { Find out which zone it lives in. }
16 2100 --
16 2101 --      IF (ResError <> noErr) |                               { If it's a MultiFinder fake DA, }
16 2102 --          IsOpen(theID) |                                   { _or if the DA is already open, }
16 2103 --          (NOT ourHeap) THEN                               { _or if it's not going in our heap }
16 2104 --          aRefNum := OpenDeskAcc(deskAccName)             { _then go ahead and open it.    }
16 2105 --      ELSE
16 2106 --          BEGIN
16 2107 --              { If we get this far, we know we have a real DA and it's going into our }
16 2108 --              { heap.  Open it, but then make sure we have enough memory to continue }
16 2109 --              { running.                                                            }
16 2110 --              FailSpaceIsLow; { In case we're already low on mem. }
16 2111 --
16 2112 --              aRefNum := OpenDeskAcc(deskAccName); { Use temporary allocation. }
16 2113 --
16 2114 --              FailSpaceIsLow; { Fail if not enough memory left. }
16 2115 --              FailNIL(drvvrH); { _or if the driver was purged to }
16 2116 --                              { _satisfy a code space requirement. }
16 2117 --          END;
16 2118 --
16 2119 --      Success(fi);
16 2120 --
16 2121 --  END { TApplication.OpenDeskAccessory };
16 2122 --
16 2123 --  ($$ MAOpen)
16 2124 --  {-----+
16 2125 --  | OpenNew |
16 2126 --  +-----+
16 2127 --  A PROCEDURE TApplication.OpenNew(itsCmdNumber: CmdNumber);
16 2128 --  VAR
16 2129 --      aDocument: TDocument;
16 2130 --      fi: FailInfo;
16 2131 --      newTitle: Str255;
16 2132 --      aWindow: TWindow;
16 2133 --
16 2134 --      {-----+
16 2135 --      | HdlopenNew |
16 2136 --      +-----+
16 2137 --      B PROCEDURE HdlopenNew(error: INTEGER; message: LONGINT);
16 2138 --      BEGIN
16 2139 --          FreeObject(aDocument);
16 2140 --          FailNewMessage(error, message, msgNewFailed);
16 2141 --      END;
16 2142 --
16 2143 --      A BEGIN
16 2144 --          aDocument := NIL;
16 2145 --          CatchFailures(fi, HdlopenNew);
16 2146 --
16 2147 --          aDocument := DoMakeDocument(KindOfDocument(itsCmdNumber, NIL));
16 2148 --          aDocument.DoInitialState;
16 2149 --          aDocument.DoMakeViews(kForDisplay);
16 2150 --          aDocument.DoMakeWindows;
16 2151 --
16 2152 --          aDocument.UntitledName(newTitle);
16 2153 --          { For MacApp 1.1, newTitle should be always <> '' }
16 2154 --          IF newTitle <> '' THEN
16 2155 --              aDocument.SetTitle(newTitle)
16 2156 --          ELSE IF aDocument.fWindowList.fSize > 0 THEN
16 2157 --              BEGIN { must set fTitle field anyways }
16 2158 --                  aWindow := TWindow(aDocument.fWindowList.First);
16 2159 --
16 2160 --                  GetWTitle(aWindow.fWmgrWindow, newTitle);
16 2161 --                  aDocument.fTitle := Copy(newTitle,
16 2162 --                      aWindow.fPreDocName,
16 2163 --                      Length(newTitle)-aWindow.fConstTitle);
16 2164 --              END;
16 2165 --
16 2166 --          AddDocument(aDocument);
16 2167 --
16 2168 --          FailSpaceIsLow; { Fail if document leaves us with no room }
16 2169 --
16 2170 --          { Don't attempt to show the windows until we're sure we won't fail }
16 2171 --          aDocument.ShowWindows;
16 2172 --
16 2173 --          Success(fi);
16 2174 --      END;
16 2175 --  END;
16 2176 --  A
16 2177 --  END;

```

```

16 2178 --
16 2179 --      {$S MAOpen}
16 2180 --      {-----+
16 2181 --      |   OpenOld   |
16 2182 --      +-----+}
16 2183 -- A  PROCEDURE TApplication.OpenOld(itsOpenCmd: CmdNumber; anAppFile: AppFile);
16 2184 --      { Called for opening a document, given its name }
16 2185 --      VAR
16 2186 --          aDocument:    TDocument;
16 2187 --          otherDoc:      TDocument;
16 2188 --          oldCodeReserve,
16 2189 --          oldMemReserve: Size;
16 2190 --          fi:            FailInfo;
16 2191 --
16 2192 --      {-----+
16 2193 --      |   Hd1OpenOld   |
16 2194 --      +-----+}
16 2195 -- B  PROCEDURE Hd1OpenOld(error: INTEGER; message: LONGINT);
16 2196 -- 0- B  BEGIN
16 2197 --      FreeObject(aDocument);
16 2198 --
16 2199 --      IF message = 0 THEN
16 2200 --          gErrorParm3 := anAppFile.fName;
16 2201 --          { Set the reserve back to where it was }
16 2202 --          SetReserveSize(oldCodeReserve, oldMemReserve);
16 2203 --          FailNewMessage(error, message, msgOpenFailed);
16 2204 -- 0 B  END;
16 2205 --
16 2206 -- 0- A  BEGIN
16 2207 --      aDocument := NIL;
16 2208 --
16 2209 --      CatchFailures(fi, Hd1OpenOld);
16 2210 --
16 2211 --      { Set reserve down a little to ensure that we can open existing documents }
16 2212 --      GetReserveSize(oldCodeReserve, oldMemReserve);
16 2213 --      SetReserveSize(oldCodeReserve, oldMemReserve DIV 2);
16 2214 --
16 2215 --      otherDoc := AlreadyOpen(anAppFile.fName, anAppFile.vRefnum);
16 2216 --      IF otherDoc <> NIL THEN
16 2217 --          otherDoc.OpenAgain(itsOpenCmd, aDocument);
16 2218 --
16 2219 --      aDocument := DoMakeDocument(KindOfDocument(itsOpenCmd, @anAppFile));
16 2220 --
16 2221 --      aDocument.ReadFromFile(anAppFile, kForDisplay);
16 2222 --      aDocument.DoMakeViews(kForDisplay);
16 2223 --      aDocument.DoMakeWindows;
16 2224 --      (** aDocument.SetTitle(aDocument.fTitle); **)
16 2225 --      { ??? The preceeding should not be necessary because ReadFromFile correctly sets
16 2226 --          fTitle, and TWindow.IWindow correctly sets the window title based on the
16 2227 --          fTitle field. (Good style?) ??? }
16 2228 --      AddDocument(aDocument);
16 2229 --
16 2230 --      FailSpaceIsLow; { Fail if the document leaves us with no memory }
16 2231 --      { Set the reserve back to where it was }
16 2232 --      SetReserveSize(oldCodeReserve, oldMemReserve);
16 2233 --
16 2234 --      { Don't attempt to show the windows until we're sure we won't fail }
16 2235 --      aDocument.ShowWindows;
16 2236 --
16 2237 --      Success(fi);
16 2238 -- 0 A  END;
16 2239 --
16 2240 --      {$S MRes}
16 2241 --      {-----+
16 2242 --      |   PerformCommand   |
16 2243 --      +-----+}
16 2244 -- A  PROCEDURE TApplication.PerformCommand(command: TCommand);
16 2245 --      VAR
16 2246 --          fi:            FailInfo;
16 2247 --          saveCmd:       BOOLEAN;
16 2248 --
16 2249 --      {-----+
16 2250 --      |   Hd1DoIt   |
16 2251 --      +-----+}
16 2252 -- B  PROCEDURE Hd1DoIt(error: INTEGER; message: LONGINT);
16 2253 --
16 2254 -- 0- B  BEGIN
16 2255 --      IF gClipClaimed THEN
16 2256 -- 1-  BEGIN
16 2257 --          SetClipView(gClipUndoView);
16 2258 --          gClipUndoView := NIL;
16 2259 --          { The newly-installed view needs to be freed also }
16 2260 --          (* SwapClipViews; { Get original back there } { would be nice but doesn't do right thing yet } *)
16 2261 -- 1-  END;
16 2262 --
16 2263 --          command.Free;
16 2264 --          gLastCommand := NIL;
16 2265 --
16 2266 --          FailNewMessage(error, message, BuildMessage(command.fCmdNumber, msgCmdErr));
16 2267 -- 0 B  END;
16 2268 --
16 2269 -- 0- A  BEGIN
16 2270 --      {$IFC qDebug}
16 2271 --      IF command = NIL THEN
16 2272 --          ProgramBreak('NIL passed to TApplication.PerformCommand')
16 2273 --      ELSE IF command = gNoChanges THEN
16 2274 --          ProgramBreak('gNoChanges passed to TApplication.PerformCommand')
16 2275 --      ELSE
16 2276 --          ($ENDC)

```

```

16 2277 --      IF command <> gLastCommand THEN
16 2278 1-      BEGIN
16 2279 --          saveCmd := command.fCausesChange OR command.fCanUndo;
16 2280 --
16 2281 --          IF saveCmd THEN
16 2282 --              CommitLastCommand;
16 2283 --
16 2284 --          command.fTarget := gTarget;
16 2285 --
16 2286 --          CatchFailures(fi, Hdldoit);
16 2287 --          IF gEventLevel = 1 THEN      { Don't unload segs if in nested event handling }
16 2288 --              UnloadAllSegments;
16 2289 --
16 2290 --          gClipClaimed := FALSE;
16 2291 --          command.DoIt;
16 2292 --          Success(fi);
16 2293 --
16 2294 --          IF saveCmd THEN
16 2295 --              BEGIN
16 2296 2-              gLastCommand := command;
16 2297 --              command.fCmdDone := TRUE;
16 2298 --              END;
16 2299 -2
16 2300 --          WITH command DO
16 2301 --              IF fCausesChange THEN { put this after .DoIt, so .DoIt can change this flag }
16 2302 --                  BEGIN
16 2303 2-                  IF fChangedDocument <> NIL THEN
16 2304 --                      WITH fChangedDocument DO
16 2305 --                          fChangeCount := fChangeCount + 1;
16 2306 --                      END;
16 2307 -2
16 2308 --                  IF NOT saveCmd THEN
16 2309 --                      command.Free;
16 2310 --                  END;
16 2311 -1
16 2312 -0 A      END;
16 2313 --
16 2314 --      {$S MARES}
16 2315 --      {-----+
16 2316 --      |   Pollevent   |
16 2317 --      +-----+}
16 2318 -- A      PROCEDURE TApplication.Pollevent;
16 2319 --          LABEL 1000;
16 2320 --
16 2321 --      VAR
16 2322 --          fi:          FailInfo;
16 2323 --          theEvent:    EventRecord;
16 2324 --          aWindow:     TWindow;
16 2325 --
16 2326 --      {-----+
16 2327 --      |   HdldPollEvt   |
16 2328 --      +-----+}
16 2329 -- B      PROCEDURE HdldPollEvt(error: INTEGER; message: LONGINT);
16 2330 0- B      BEGIN
16 2331 --          {$IFC qDebug}
16 2332 --              Writeln; { add a blank line after all the messages from Failure }
16 2333 --          {$ENDC}
16 2334 --          IF error <> noErr THEN
16 2335 1-              BEGIN
16 2336 --                  IF gEventLevel = 1 THEN      { Don't unload segs if nested event handling. }
16 2337 --                      UnloadAllSegments;
16 2338 --                  ShowError(error, message);
16 2339 -1
16 2340 --                  HiliteMenu(0);                { Make sure menu isn't left highlighted. }
16 2341 --                  GOTO 1000;                    { Keep the application running. }
16 2342 -0 B      END;
16 2343 --
16 2344 --
16 2345 0- A      BEGIN
16 2346 --          gEventLevel := gEventLevel + 1;
16 2347 --
16 2348 --          {$IFC qDebug}
16 2349 --              IF gTarget = NIL THEN
16 2350 --                  Writeln('Serious Error!!! in TApplication.Pollevent: target = NIL');
16 2351 --              {$ENDC}
16 2352 --
16 2353 --          CatchFailures(fi, HdldPollEvt);
16 2354 --
16 2355 --          IF GetEvent(gMainEventMask, gIdleFreq, gCursorRgn, theEvent) THEN
16 2356 1-              BEGIN
16 2357 --                  IF gIdlePhase <> idleBegin THEN
16 2358 2-                      BEGIN
16 2359 --                          Idle(idleEnd);
16 2360 --                          gIdlePhase := idleBegin;
16 2361 -2
16 2362 --                          HandleEvent(theEvent);
16 2363 --                          HiliteMenu(0);
16 2364 --                      {$IFC qDebug}
16 2365 --                          gErrorParm3 := '?????'; { to prevent anyone from using old values }
16 2366 --                      {$ENDC}
16 2367 -1
16 2368 --                      END
16 2369 --                  ELSE { idle }
16 2370 1-                      BEGIN
16 2371 --                          Idle(gIdlePhase);
16 2372 --                          gIdlePhase := idleContinue;
16 2373 -1
16 2374 --                          END;
16 2375 --                  { The desk scrap may have been changed by use of Cmd-X or Cmd-C in

```

```

16 2376 --      desk accessories. )
16 2377 --      IF gSysWindowActive THEN
16 2378 --          CheckDeskScrap;
16 2379 --
16 2380 --      Success(fi);
16 2381 --
16 2382 --      1000:
16 2383 --      gEventLevel := gEventLevel - 1;
16 2384 --0 A  END;
16 2385 --
16 2386 --      {$S MAREs}
16 2387 --      {-----+
16 2388 --      | PostHandleEvent |
16 2389 --      +-----+}
16 2390 -- A  PROCEDURE TApplication.PostHandleEvent (VAR theEventInfo: EventInfo);
16 2391 --
16 2392 --      VAR
16 2393 --          sysWindowAct:    BOOLEAN;
16 2394 --          perm:            BOOLEAN;
16 2395 --
16 2396 --0 A  BEGIN
16 2397 --      IF gRedrawMenuBar THEN      { application wants menus to be setup }
16 2398 --          SetupTheMenus
16 2399 --      ELSE
16 2400 --          gMenusAreSetup := FALSE; { menus may need to be setup }
16 2401 --
16 2402 --      perm := PermAllocation(FALSE);
16 2403 --      {$IFC qDebug}
16 2404 --      IF perm THEN
16 2405 --          ProgramBreak('The permanent flag was left TRUE.');
```

```

16 2475 --      Success(fi);
16 2476 --
16 2477 --      aDocument.Free;
16 2478 --      UnloadAllSegments;
16 2479 --      PrintDocument := proceed;
16 2480 -- A   END;
16 2481 --
16 2482 --      {$S MAClipboard}
16 2483 --      {-----+
16 2484 --      |   ReadFromDeskScrap   |
16 2485 --      +-----}
16 2486 -- A   PROCEDURE TApplication.ReadFromDeskScrap;
16 2487 --      LABEL 1000;
16 2488 --      VAR
16 2489 --          aViewForClipboard: TView;
16 2490 --          fi:                FailInfo;
16 2491 --
16 2492 --      {-----+
16 2493 --      |   Hd1MakeViewForAlienClipbd |
16 2494 --      +-----}
16 2495 -- B   PROCEDURE Hd1MakeViewForAlienClipbd(error: OSErr; message: LONGINT);
16 2496 -- B   BEGIN
16 2497 --
16 2498 --          aViewForClipboard := gClipOrphanage;
16 2499 --          IF message = 0 THEN
16 2500 --              message := msgImportClipFailed;
16 2501 --              ShowError(error, message);
16 2502 --              GOTO 1000;
16 2503 -- B   END;
16 2504 --
16 2505 -- A   BEGIN
16 2506 --      CatchFailures(fi, Hd1MakeViewForAlienClipbd);
16 2507 --      aViewForClipboard := MakeViewForAlienClipboard;
16 2508 --      FailNil(aViewForClipboard);
16 2509 --      Success(fi);
16 2510 --      1000:
16 2511 --      ClaimClipboard(aViewForClipboard);
16 2512 -- A   END;
16 2513 --
16 2514 --      {$S MAREs}
16 2515 --      {-----+
16 2516 --      |   RegainControl   |
16 2517 --      +-----}
16 2518 -- A   PROCEDURE TApplication.RegainControl(checkClipboard: BOOLEAN);
16 2519 -- A   BEGIN
16 2520 --      BusyActivate(TRUE);
16 2521 --      IF checkClipboard THEN
16 2522 --          CheckDeskScrap;
16 2523 -- A   END;
16 2524 --
16 2525 --      {$IFC qDebug}
16 2526 --      {$S MADEbug}
16 2527 --      { Debugging procedure: given an EventRecord, prints out information about the event. }
16 2528 --      {$IFC qTrace}{$D+}{$SENDC}
16 2529 --      {-----+
16 2530 --      |   ReportEvent   |
16 2531 --      +-----}
16 2532 -- A   PROCEDURE TApplication.ReportEvent(VAR theEventInfo: EventInfo);
16 2533 --      VAR
16 2534 --          ch:    INTEGER;
16 2535 --          cap:   INTEGER;
16 2536 --          wName: Str255;
16 2537 --          mods:  STRING[10];
16 2538 --
16 2539 -- A   BEGIN
16 2540 --      WITH theEventInfo, thePEvent^ DO
16 2541 --          BEGIN
16 2542 --              Write('t = ', when);
16 2543 --              mods := ' ';
16 2544 --              { 1234567890 };
16 2545 --              IF theControlKey THEN
16 2546 --                  mods[2] := 'C';
16 2547 --              IF theOptionKey THEN
16 2548 --                  mods[3] := 'O';
16 2549 --              IF theAlphaLock THEN
16 2550 --                  mods[4] := 'L';
16 2551 --              IF theShiftKey THEN
16 2552 --                  mods[5] := 'S';
16 2553 --              IF theCmdKey THEN
16 2554 --                  mods[6] := 'C';
16 2555 --              IF theBtnState THEN
16 2556 --                  mods[7] := 'M';
16 2557 --              IF what = activateEvt THEN
16 2558 --                  IF BAND(modifiers, activeFlag) <> 0 THEN
16 2559 --                      mods[8] := 'A'
16 2560 --                  ELSE
16 2561 --                      mods[8] := 'D';
16 2562 --              Write(mods);
16 2563 --
16 2564 --          CASE what OF
16 2565 --              nullEvent:
16 2566 --                  WriteLn('nullEvent  ');
16 2567 --              mouseDown, mouseUp:
16 2568 --                  BEGIN
16 2569 --                      IF what = mouseDown THEN
16 2570 --                          Write('mouseDown  ')
16 2571 --                      ELSE
16 2572 --                          Write('mouseUp  ');
16 2573 --                  WriteLn('@ ('', where.h:1, ', ', where.v:1, '));

```



```

16 2574 --      END;
16 2575 --      keyDown, autoKey, keyUp:
16 2576 --      BEGIN
16 2577 --          IF what = keyDown THEN
16 2578 --              Write('keyDown ')
16 2579 --          ELSE IF what = autoKey THEN
16 2580 --              Write('autoKey ')
16 2581 --          ELSE
16 2582 --              Write('keyUp ');
16 2583 --
16 2584 --          ch := BAND(message, charCodeMask);
16 2585 --          cap := BSR(message, 8);
16 2586 --
16 2587 --          IF (ch >= $20) AND (ch <= $D8) AND (ch <> $7F) THEN
16 2588 --              Write(' ', CHR(ch), ' ');
16 2589 --          ELSE
16 2590 --              Write(' ');
16 2591 --
16 2592 --          WriteLn('(', ch:1, '/', cap:1, ')');
16 2593 --      END;
16 2594 --      updateEvt, activateEvt:
16 2595 --      BEGIN
16 2596 --          IF what = updateEvt THEN
16 2597 --              Write('updateEvt ')
16 2598 --          ELSE
16 2599 --              Write('activateEvt ');
16 2600 --
16 2601 --          wName := WindowPeek(message)^.titleHandle^^;
16 2602 --          WriteLn(' ', wName, ' ');
16 2603 --      END;
16 2604 --      diskEvt:
16 2605 --          WriteLn('diskEvt ', 'd = ', LoWord(message):1, ' e = ', HiWord(message):1);
16 2606 --      networkEvt:
16 2607 --      BEGIN
16 2608 --          Write ('networkEvt ');
16 2609 --          WritePtr(message);
16 2610 --          WriteLn;
16 2611 --      END;
16 2612 --      driverEvt:
16 2613 --          WriteLn('driverEvt ', message);
16 2614 --      applEvt:
16 2615 --          WriteLn('applEvt ', message);
16 2616 --      app2Evt:
16 2617 --          WriteLn('app2Evt ', message);
16 2618 --      app3Evt:
16 2619 --          WriteLn('app3Evt ', message);
16 2620 --      app4Evt:
16 2621 --          IF BAND(message, $FF000000) = kSuspendOrResume THEN
16 2622 --              IF ODD(message) THEN
16 2623 --                  WriteLn('switch in ', message)
16 2624 --              ELSE
16 2625 --                  WriteLn('switch out ', message)
16 2626 --              ELSE IF BAND(message, $FF000000) = kMouseMovedMessage THEN
16 2627 --                  WriteLn('mouse moved ', message)
16 2628 --              ELSE
16 2629 --                  WriteLn('app4Evt ', message);
16 2630 --      OTHERWISE
16 2631 --          WriteLn('??? unknown = ', what:1, ' ', message:1);
16 2632 --      END;
16 2633 --      END;
16 2634 -- 0 A      END;
16 2635 --      {$IFC qTrace}{$D+}{$ENDC}
16 2636 --      {$ENDC qDebug}
16 2637 --
16 2638 --      {$IFC qDebug}
16 2639 --      {$S MAdDebug}
16 2640 --      {$IFC qTrace}{$D+}{$ENDC}
16 2641 --      { Debugging procedure: given the result of FindWindow, WriteLn the location of the mouse. }
16 2642 --      {-----+
16 2643 --      | ReportMouseDown |
16 2644 --      +-----}
16 2645 -- 0 A      PROCEDURE TApplication.ReportMouseDown(where: INTEGER);
16 2646 --      VAR
16 2647 --          str: Str255;
16 2648 -- 0 A      BEGIN
16 2649 --          CASE where OF
16 2650 --              inMenuBar:
16 2651 --                  str := 'inMenuBar';
16 2652 --              inSysWindow:
16 2653 --                  str := 'inSysWindow';
16 2654 --              inDrag:
16 2655 --                  str := 'inDrag';
16 2656 --              inGrow:
16 2657 --                  str := 'inGrow';
16 2658 --              inGoAway:
16 2659 --                  str := 'inGoAway';
16 2660 --              inContent:
16 2661 --                  str := 'inContent';
16 2662 --              inZoomIn:
16 2663 --                  str := 'inZoomIn';
16 2664 --              inZoomOut:
16 2665 --                  str := 'inZoomOut';
16 2666 --      OTHERWISE
16 2667 --          BEGIN
16 2668 --              WriteLn(' ':30, where:1);
16 2669 --              str := 'Mouse clicked in an unknown place.'
16 2670 --          END;
16 2671 --      END; { CASE }
16 2672 --      WriteLn(' ':30, str);

```

```

16 2673 -0 A  END;
16 2674 --  { $IFC qTrace } { $D++ } { $ENDC }
16 2675 --  { $ENDC qDebug }
16 2676 --
16 2677 --  { $$ Main } { must be in main segment }
16 2678 --  {-----+
16 2679 --  | Run |
16 2680 --  +-----+
16 2681 -- A  PROCEDURE TApplication.Run;
16 2682 --  VAR
16 2683 --      findSeg: INTEGER;
16 2684 0- A  BEGIN
16 2685 --      UnloadAllSegments;
16 2686 --
16 2687 --      FailSpaceIsLow; { make sure we have enough memory to continue }
16 2688 --
16 2689 --      ginitialized := TRUE; { was set FALSE in InitToolBox }
16 2690 --      SetInitHandler(NIL);
16 2691 --
16 2692 --      IF NOT gFinderPrinting THEN
16 2693 --          LaunchClipboard;
16 2694 --      UnloadAllSegments;
16 2695 --
16 2696 --      findSeg := GetSegNumber(@FinderSegProc);
16 2697 --      IF gFinderPrinting THEN
16 2698 --          SetResidentSegment(findSeg, TRUE);
16 2699 --
16 2700 --      HandleFinderRequest;
16 2701 --
16 2702 --      IF gFinderPrinting THEN
16 2703 --          SetResidentSegment(findSeg, FALSE);
16 2704 --      UnloadAllSegments;
16 2705 --
16 2706 --      IF NOT gFinderPrinting THEN
16 2707 1-          BEGIN
16 2708 --          gRedrawMenuBar := TRUE; { force setup/redraw of menu bar }
16 2709 --          SetupTheMenus;
16 2710 --
16 2711 --          MainEventLoop;
16 2712 --
16 2713 --          AboutToLoseControl(TRUE);
16 2714 --          END
16 2715 --      ELSE
16 2716 --          Close; { Close is always called when quitting app }
16 2717 --
16 2718 --
16 2719 --      { $IFC qDebug }
16 2720 --      { See if previous max. resource usage has been exceeded by the termi-
16 2721 --      nation code and resources. }
16 2722 --      CheckRsrcUsage;
16 2723 --      { $ENDC }
16 2724 --
16 2725 --      { We must call CleanupMacApp here; if we wait to fall thru to the end of the
16 2726 --      main program, A5 has been invalidated and we can't refer to any globals. }
16 2727 --      CleanupMacApp;
16 2728 -0 A  END;
16 2729 --
16 2730 --  { $$ MArEs }
16 2731 --  {-----+
16 2732 --  | SelectWMgrWindow |
16 2733 --  +-----+
16 2734 -- A  PROCEDURE TApplication.SelectWMgrWindow (aWMgrWindow: WindowPtr);
16 2735 --
16 2736 0- A  BEGIN
16 2737 --      SelectWindow(aWMgrWindow); { Simply call the toolbox to select it. }
16 2738 --      gLastClickPart := inDesk; { Make sure previous mouse clicks are not }
16 2739 --      { are not considered part of a multi-click. }
16 2740 -0 A  END;
16 2741 --
16 2742 --  { $$ MAClipboard }
16 2743 --  {-----+
16 2744 --  | SetClipView |
16 2745 --  +-----+
16 2746 -- A  PROCEDURE TApplication.SetClipView (clipView: TView);
16 2747 --
16 2748 --  VAR
16 2749 --      theSuperView: TView;
16 2750 --
16 2751 -- B  PROCEDURE RemoveView (aView: TView);
16 2752 --
16 2753 0- B  BEGIN
16 2754 --      theSuperView.RemoveSubView(aView);
16 2755 -0 B  END;
16 2756 --
16 2757 0- A  BEGIN
16 2758 --      IF gClipWindow <> NIL THEN
16 2759 1-          BEGIN
16 2760 --          IF gClipWindow.CountSubViews > 0 THEN
16 2761 --              theSuperView := TView(gClipWindow.fSubViews.First)
16 2762 --          ELSE
16 2763 --              theSuperView := gClipWindow;
16 2764 --              theSuperView.EachSubView(RemoveView);
16 2765 --              theSuperView.AddSubView(clipView);
16 2766 --              clipView.fSuperView := theSuperView;
16 2767 --              clipView.SuperViewChangedSize(gZeroVPt, FALSE);
16 2768 --              clipView.RevealTop(kDontRedraw);
16 2769 --              gClipWindow.ForceRedraw;
16 2770 --              gClipWindow.fTarget := gClipWindow;
16 2771 --              gClipWrittenToDeskScrap := clipView - gClipOrphanage;

```

```

16 2772 -1      END
16 2773 --      ELSE
16 2774 1-      BEGIN
16 2775 --      ($IFC qDebug)
16 2776 --      ProgramBreak('SetClipView in absence of gClipWindow');
16 2777 --      ($ENDC)
16 2778 -1      END;
16 2779 --
16 2780 --      gClipView := clipView;
16 2781 -0 A  END;
16 2782 --
16 2783 --      ($$ MRes)
16 2784 --      {-----+
16 2785 --      | SetTarget |
16 2786 --      +-----+}
16 2787 -- A  PROCEDURE TApplication.SetTarget (newTarget: TEvtHandler);
16 2788 --
16 2789 0- A  BEGIN
16 2790 --      gTarget.InstallSelection(TRUE, FALSE);
16 2791 --      newTarget.InstallSelection(FALSE, TRUE);
16 2792 --      gTarget := newTarget;
16 2793 --      gIdleFreq := 0;      { Make sure we idle ASAP because there's a new target. }
16 2794 --      SetEmptyRgn(gCursorRgn); { Make sure cursor region gets set as well. }
16 2795 -0 A  END { TApplication.SetTarget };
16 2796 --
16 2797 --      ($$ MRes)
16 2798 --      {-----+
16 2799 --      | SetUndoText |
16 2800 --      +-----+}
16 2801 -- A  PROCEDURE TApplication.SetUndoText(cmdDone: BOOLEAN; aCmdNumber: CmdNumber);
16 2802 --      VAR
16 2803 --      newMenuState: INTEGER;
16 2804 --      undoName: Str255;
16 2805 --      cmdName: Str255;
16 2806 --      preCmdName: INTEGER;
16 2807 --      constChars: INTEGER;
16 2808 0- A  BEGIN
16 2809 --      IF (gUndoState <> cmdDone) OR (gUndoCmd <> aCmdNumber) THEN
16 2810 1-      BEGIN
16 2811 --      IF aCmdNumber = cCantUndo THEN
16 2812 --      newMenuState := bzCantUndo
16 2813 --      ELSE IF cmdDone THEN
16 2814 --      newMenuState := bzUndo
16 2815 --      ELSE
16 2816 --      newMenuState := bzRedo;
16 2817 --
16 2818 --      GetIndString(undoName, kIDBuzzString, newMenuState);
16 2819 --      IF ParseTitleTemplate(undoName, preCmdName, constChars) THEN
16 2820 2-      BEGIN
16 2821 --      IF (aCmdNumber = cNoCommand) OR (aCmdNumber = cCantUndo) THEN
16 2822 --      cmdName := ''
16 2823 --      ELSE
16 2824 --      CmdToName(aCmdNumber, cmdName);
16 2825 --      IF SubstituteInTitle(undoName, cmdName, preCmdName, constChars) THEN;
16 2826 -2      END;
16 2827 --
16 2828 --      SetCmdName(cUndo, undoName);
16 2829 --
16 2830 --      gUndoState := cmdDone;
16 2831 --      gUndoCmd := aCmdNumber;
16 2832 -1      END;
16 2833 -0 A  END;
16 2834 --
16 2835 --      ($$ MRes)
16 2836 --      {-----+
16 2837 --      | SetupTheMenus |
16 2838 --      +-----+}
16 2839 -- A  PROCEDURE TApplication.SetupTheMenus;
16 2840 --
16 2841 --      {-----+
16 2842 --      | DoSetup |
16 2843 --      +-----+}
16 2844 -- B  PROCEDURE DoSetup;
16 2845 --
16 2846 --      VAR
16 2847 --      appleMenu: MenuHandle;
16 2848 --      undoState: BOOLEAN;
16 2849 --      undoCmd: CmdNumber;
16 2850 --
16 2851 0- B  BEGIN
16 2852 --      gGotClipType := FALSE;
16 2853 --
16 2854 --      gTarget.DoSetupMenus;      { Setup menus relevant to target chain }
16 2855 --
16 2856 --      { Set up the menu commands that are not dependent on the target chain... }
16 2857 --
16 2858 --      appleMenu := GetMHandle(mApple);
16 2859 --      WITH appleMenu^^ DO
16 2860 --      IF ODD(enableFlags) = InModalState THEN
16 2861 1-      BEGIN
16 2862 --      enableFlags := BXOR(enableFlags, 1);
16 2863 --      gRedrawMenuBar := TRUE;
16 2864 -1      END;
16 2865 --
16 2866 --      undoState := kShowCantUndo;      { Set the Undo menu defaults. }
16 2867 --      undoCmd := cCantUndo;
16 2868 --      IF gSysWindowActive THEN
16 2869 1-      BEGIN
16 2870 --      undoState := kShowUndo;

```

```

16 2871 --      undoCmd := cNoCommand;
16 2872 --      END
16 2873 --      ELSE IF gLastCommand <> NIL THEN
16 2874 --          WITH gLastCommand DO
16 2875 --              IF fCanUndo THEN
16 2876 --                  BEGIN
16 2877 --                      IF fCmdDone THEN
16 2878 --                          undoState := kShowUndo
16 2879 --                      ELSE
16 2880 --                          undoState := kShowRedo;
16 2881 --                          undoCmd := fCmdNumber;
16 2882 --                      { Enable Undo only if the last command was not document-specific
16 2883 --                        or the document changed is the current document. }
16 2884 --                      Enable(cUndo, (fChangedDocument = NIL) OR (fChangedDocument = gDocument));
16 2885 --                      END;
16 2886 --                      SetUndoText(undoState, undoCmd);
16 2887 --
16 2888 --
16 2889 --      ($IFC qDebug)
16 2890 --          EnableCheck(cExperimenting, TRUE, gExperimenting);
16 2891 --          EnableCheck(cReportEvt, TRUE, gReportEvt);
16 2892 --          EnableCheck(cDebugPrinting, TRUE, gDebugPrinting);
16 2893 --          EnableCheck(cReportMenuChoices, TRUE, gReportMenuChoices);
16 2894 --          EnableCheck(cIntenseDebugging, TRUE, gIntenseDebugging);
16 2895 --          Enable(cIdentifySoftware, TRUE);
16 2896 --
16 2897 --      IF gFrontWindow <> NIL THEN
16 2898 --          BEGIN
16 2899 --              Enable(cModalToggle, TRUE);
16 2900 --              SetMenuState(cModalToggle, kDebugBuzzStrings, bzMakeModal, bzMakeModeless,
16 2901 --                gFrontWindow.fIsModal);
16 2902 --              Enable(cRefreshFrontWindow, TRUE);
16 2903 --              Enable(cDoFirstClick, TRUE);
16 2904 --              SetMenuState(cDoFirstClick, kDebugBuzzStrings, bzDoFirstClick,
16 2905 --                bzDontDoFirstClick, gFrontWindow.fDoFirstClick);
16 2906 --          END;
16 2907 --      ($ENDC)
16 2908 --      ($IFC qTrace)
16 2909 --          EnableCheck(cTraceSetupMenus, TRUE, gTraceSetupMenus);
16 2910 --          EnableCheck(cTraceIdle, TRUE, gTraceIdle);
16 2911 --      ($ENDC)
16 2912 --
16 2913 --      IF NOT gSysWindowActive THEN
16 2914 --          Enable(cPaste, gGotClipType);
16 2915 --      END ( DoSetup );
16 2916 --
16 2917 -- 0- A BEGIN
16 2918 --      IF NOT gMenusAreSetup OR gRedrawMenuBar THEN
16 2919 --          PerformMenuSetup(DoSetup);
16 2920 -- 0 A END;
16 2921 --
16 2922 --      ($S MAOpen)
16 2923 --      {-----+
16 2924 --      |   SFGGetParms
16 2925 --      +-----}
16 2926 -- 0 A PROCEDURE TApplication.SFGGetParms(itsCmdNumber: CmdNumber; VAR dlgID: INTEGER;
16 2927 --      VAR where: Point; VAR fileFilter, dlgHook, filterProc: ProcPtr;
16 2928 --      typeList: HTypeList);
16 2929 --      CONST
16 2930 --          kStdSFWhere = 100 * $10000 + 100;
16 2931 --
16 2932 -- 0- A BEGIN
16 2933 --          dlgID := getDlgID;
16 2934 --          where := Point(kStdSFWhere);
16 2935 --          fileFilter := NIL;
16 2936 --          dlgHook := NIL;
16 2937 --          filterProc := NIL;
16 2938 --          SetHandleSize(Handle(typeList), 4);
16 2939 --          typeList^[1] := gMainFileType;
16 2940 -- 0 A END;
16 2941 --
16 2942 --      ($S MAError)
16 2943 --      {-----+
16 2944 --      |   ShowError
16 2945 --      +-----}
16 2946 -- 0 A PROCEDURE TApplication.ShowError(error: OSErr; message: LONGINT);
16 2947 -- 0- A BEGIN
16 2948 --          ErrorAlert(error, message);
16 2949 -- 0 A END;
16 2950 --
16 2951 --      ($S Main)
16 2952 --      {-----+
16 2953 --      |   SpaceIsLow
16 2954 --      +-----}
16 2955 -- 0 A PROCEDURE TApplication.SpaceIsLow;
16 2956 --      VAR
16 2957 --          now: LONGINT;
16 2958 -- 0- A BEGIN
16 2959 --          IF gEventLevel = 1 THEN      { Don't unload segs if nested event handling }
16 2960 --              UnloadAllSegments;
16 2961 --
16 2962 --          { Show 'space is low' alert only after ever gLowSpaceInterval ticks. }
16 2963 --          IF gLowSpaceInterval > 0 THEN
16 2964 --              BEGIN
16 2965 --                  now := TickCount;
16 2966 --                  IF now > gNextSpaceMsg THEN
16 2967 --                      BEGIN
16 2968 --                          StdAlert(phSpaceIsLow);
16 2969 --                          gNextSpaceMsg := now + gLowSpaceInterval;

```

```

16 2970 --2          END;
16 2971 --1          END;
16 2972 --0 A  END;
16 2973 --
16 2974 --  { $$ MAClipboard }
16 2975 --  {-----+
16 2976 --  |   SwapClipViews   |
16 2977 --  +-----}
16 2978 -- A  PROCEDURE TApplication.SwapClipViews;
16 2979 --  VAR
16 2980 --      tempClipView: TView;
16 2981 --0- A  BEGIN
16 2982 --      tempClipView := gClipUndoView;
16 2983 --      gClipUndoView := gClipView;
16 2984 --
16 2985 --      IF tempClipView <> NIL THEN
16 2986 --          SetClipView(tempClipView)  { Installs old Undo clipboard as current clipboard }
16 2987 --  { $IFC qDebug }
16 2988 --      ELSE
16 2989 --          ProgramBreak('SwapClipViews finds undo clipboard was NIL')
16 2990 --  { $ENDC }
16 2991 --      ;
16 2992 --0 A  END;
16 2993 --
16 2994 --  { $$ MAREs }
16 2995 --  {-----+
16 2996 --  |   TrackCursor   |
16 2997 --  +-----}
16 2998 -- A  FUNCTION TApplication.TrackCursor: BOOLEAN;
16 2999 --
16 3000 --  VAR
16 3001 --      globalMouse:    Point;
16 3002 --      cursorIsSet:    BOOLEAN;
16 3003 --      aWMgrWindow:    WindowPtr;
16 3004 --      cursorWindow:   TWindow;
16 3005 --      cursorView:     TView;
16 3006 --      windowVPt:      VPoint;
16 3007 --      windowBounds:   Rect;
16 3008 --      r:              Rect;
16 3009 --      haveCursorRgn:  BOOLEAN;
16 3010 --
16 3011 --0- A  BEGIN
16 3012 --      IF (gFrontWindow = NIL) OR gInBackground THEN
16 3013 --          EXIT(TrackCursor);
16 3014 --
16 3015 --      GetMouse(globalMouse);
16 3016 --      LocalToGlobal(globalMouse);
16 3017 --
16 3018 --      IF PtInRgn(globalMouse, gCursorRgn) THEN
16 3019 --1-      BEGIN
16 3020 --  { $IFC qDebug }
16 3021 --      IF gIntenseDebugging AND gTraceIdle THEN
16 3022 --          WRITELN('cursor is in cursor region');
16 3023 --  { $ENDC }
16 3024 --      EXIT(TrackCursor);
16 3025 --1-      END;
16 3026 --
16 3027 --      IF (FindWindow(globalMouse, aWMgrWindow) = inContent) THEN
16 3028 --1-      BEGIN
16 3029 --          { The cursor is in a window--for our purposes, it must be the front
16 3030 --            window or the window must handle first clicks }
16 3031 --          cursorWindow := WMgrToWindow(aWMgrWindow);
16 3032 --          IF (cursorWindow <> gFrontWindow) & (NOT cursorWindow.fDoFirstClick) THEN
16 3033 --              cursorWindow := NIL;
16 3034 --1-          END
16 3035 --      ELSE
16 3036 --          cursorWindow := NIL;
16 3037 --
16 3038 --      SetEmptyRgn(gCursorRgn);
16 3039 --      haveCursorRgn := FALSE;
16 3040 --      cursorIsSet := FALSE;
16 3041 --      IF cursorWindow <> NIL THEN
16 3042 --1-      BEGIN
16 3043 --          cursorWindow.GetGlobalBounds(windowBounds);
16 3044 --          windowVPt.h := globalMouse.h - windowBounds.left;
16 3045 --          windowVPt.v := globalMouse.v - windowBounds.top;
16 3046 --          cursorView := cursorWindow.HandleCursor(windowVPt, gCursorRgn);
16 3047 --          IF cursorView <> NIL THEN
16 3048 --2-      BEGIN
16 3049 --          cursorIsSet := TRUE;
16 3050 --
16 3051 --          IF (NOT EmptyRgn(gCursorRgn)) & (cursorView.CountSubViews = 0) THEN
16 3052 --3-      BEGIN
16 3053 --              { Intersect with viewed rect }
16 3054 --              cursorView.GetVisibleRect(r);
16 3055 --  { $IFC qDebug }
16 3056 --          UseTempRgn('TApplication.TrackCursor');
16 3057 --  { $ENDC }
16 3058 --          RectRgn(gTempRgn, r);
16 3059 --          SectRgn(gTempRgn, gCursorRgn, gCursorRgn);
16 3060 --  { $IFC qDebug }
16 3061 --          DoneWithTempRgn;
16 3062 --  { $ENDC }
16 3063 --          { Convert gCursorRgn from view coords to global coords }
16 3064 --          WITH thePort^.portRect DO
16 3065 --              OffsetRgn(gCursorRgn, windowBounds.left - left,
16 3066 --                      windowBounds.top - top);
16 3067 --          haveCursorRgn := TRUE;
16 3068 --3-      END;

```

```

16 3069 --2          END;
16 3070 --1          END;
16 3071 --
16 3072 --      IF NOT haveCursorRgn THEN
16 3073 --          WITH globalMouse DO
16 3074 --              SetRectRgn(gCursorRgn, h, v, h + 1, v + 1);
16 3075 --
16 3076 --      ($IFC qDebug)
16 3077 --          IF gIntenseDebugging AND gTraceIdle THEN
16 3078 --              IF gCursorRgn = NIL THEN
16 3079 --                  WRITELN('gCursorRgn is NIL')
16 3080 --              ELSE
16 3081 --                  BEGIN
16 3082 --                      HLock(Handle(gCursorRgn));
16 3083 --                      WrLblRect('gCursorRgn', gCursorRgn^^.RgnBBox);
16 3084 --                      WRITELN;
16 3085 --                      HUnlock(Handle(gCursorRgn));
16 3086 --                  END;
16 3087 --      ($ENDC)
16 3088 --
16 3089 --      IF NOT cursorIsSet THEN
16 3090 --          SetCursor(arrow);
16 3091 --          TrackCursor := cursorIsSet;
16 3092 --0 A  END;
16 3093 --
16 3094 --      ($S MADoCommand)
16 3095 --      {-----+
16 3096 --      |   TrackMouse   |
16 3097 --      +-----+}
16 3098 -- A  FUNCTION TApplication.TrackMouse (globalMouse, hysteresis: Point; theCommand: TCommand): TCommand;
16 3099 --
16 3100 --      VAR
16 3101 --          tracker:          TCommand;
16 3102 --          view:             TView;
16 3103 --          scroller:         TScroller;
16 3104 --          gotATracker:       BOOLEAN;
16 3105 --          theQDMouse:        Point;
16 3106 --          theMouse:          VPoint;
16 3107 --          anchorPoint:       VPoint;
16 3108 --          previousPoint:     VPoint;
16 3109 --          peekEvent:         EventRecord;
16 3110 --          movedOnce:         BOOLEAN;
16 3111 --          amtMoved:          VPoint;
16 3112 --          didMove:           BOOLEAN;
16 3113 --          delta:             VPoint;
16 3114 --          didScroll:         BOOLEAN;
16 3115 --          currTranslation:    VPoint;
16 3116 --          viewExtent:        VRect;
16 3117 --          autoscrollLimit:   VRect;
16 3118 --
16 3119 --      {-----+
16 3120 --      |   SetupFocus   |
16 3121 --      +-----+}
16 3122 -- B  PROCEDURE SetupFocus;
16 3123 --
16 3124 --      VAR
16 3125 --          wmgrPort:          GrafPtr;
16 3126 --      BEGIN
16 3127 --          IF view <> NIL THEN
16 3128 --              BEGIN
16 3129 --                  IF view.Focus THEN
16 3130 --                      BEGIN
16 3131 --                          GetFocus;
16 3132 --                      IF scroller <> NIL THEN
16 3133 --                          BEGIN
16 3134 --                              scroller.GetExtent(autoscrollLimit);
16 3135 --                          ($IFC FALSE)
16 3136 --                              AddVPt(fAutoScrollMargin.topLeft, autoscrollLimit.topLeft);
16 3137 --                              AddVPt(fAutoScrollMargin.botRight, autoscrollLimit.botRight);
16 3138 --                          ($ENDC)
16 3139 --                              currTranslation := scroller.fTranslation;
16 3140 --                              END;
16 3141 --                          END
16 3142 --                      ($IFC qDebug)
16 3143 --                      ELSE
16 3144 --                          ProgramBreak('TApplication.TrackMouse: Unable to focus view.')
16 3145 --                      ($ENDC)
16 3146 --                      ;
16 3147 --                      END
16 3148 --                  ELSE
16 3149 --                      { focus on the desktop }
16 3150 --                      BEGIN
16 3151 --                          GetWmgrPort(wmgrPort);
16 3152 --                          SetPort(wmgrPort);
16 3153 --                          ClipRect(GetGrayRgn^^.rgnBBox);
16 3154 --                          gFocusedView := NIL;
16 3155 --                          GetFocus;
16 3156 --                      END;
16 3157 --          END;
16 3158 --      ($ENDC)
16 3159 --      {-----+
16 3160 --      |   DoFocus   |
16 3161 --      +-----+}
16 3162 -- B  PROCEDURE DoFocus;
16 3163 --      BEGIN
16 3164 --          IF (scroller <> NIL) & ((currTranslation.h <> scroller.fTranslation.h) OR
16 3165 --              (currTranslation.v <> scroller.fTranslation.v)) THEN
16 3166 --              SetupFocus
16 3167 --          ELSE
16 3168 --              SetFocus;

```

```

16 3168 --0 B      END;
16 3169 --
16 3170 --      {-----+
16 3171 --      |   InstallTracker   |
16 3172 --      +-----+
16 3173 -- B      PROCEDURE InstallTracker(newTracker: TCommand);
16 3174 --0 B      BEGIN
16 3175 --          tracker := newTracker;
16 3176 --          gotATracker := (tracker <> gNoChanges);
16 3177 --          IF gotATracker THEN
16 3178 --              BEGIN
16 3179 --                  view := tracker.fView;
16 3180 --                  scroller := tracker.fScroller;
16 3181 --                  IF view <> NIL THEN
16 3182 --                      view.GetExtent(viewExtent);
16 3183 --                  END;
16 3184 --0 B      END;
16 3185 --
16 3186 --      {-----+
16 3187 --      |   FeedbackOnce     |
16 3188 --      +-----+
16 3189 -- B      PROCEDURE FeedbackOnce(turnItOn, mouseDidMove: BOOLEAN);
16 3190 --0 B      BEGIN
16 3191 --          IF gotATracker THEN
16 3192 --              BEGIN
16 3193 --                  PenNormal;
16 3194 --                  PenMode(PatXOR);
16 3195 --                  tracker.TrackFeedback(anchorPoint, previousPoint, turnItOn, mouseDidMove);
16 3196 --              END;
16 3197 --0 B      END;
16 3198 --
16 3199 --      {-----+
16 3200 --      |   ConstrainOnce    |
16 3201 --      +-----+
16 3202 -- B      PROCEDURE ConstrainOnce; { ??? fold this into TrackOnce ??? }
16 3203 --0 B      BEGIN
16 3204 --          IF gotATracker THEN
16 3205 --              BEGIN
16 3206 --                  IF tracker.fViewConstrain & (view <> NIL) THEN
16 3207 --                      PinVRect(viewExtent, theMouse);
16 3208 --                  IF tracker.fConstrainsMouse THEN
16 3209 --                      tracker.TrackConstrain(anchorPoint, previousPoint, theMouse);
16 3210 --                  END;
16 3211 --0 B      END;
16 3212 --
16 3213 --      {-----+
16 3214 --      |   TrackOnce        |
16 3215 --      +-----+
16 3216 -- B      PROCEDURE TrackOnce(aTrackPhase: TrackPhase; didMouseMove: BOOLEAN);
16 3217 --      VAR
16 3218 --          newTracker: TCommand;
16 3219 --0 B      BEGIN
16 3220 --      {$IFC qDebug}
16 3221 --          IF tracker = NIL THEN
16 3222 --              BEGIN
16 3223 --                  ProgramBreak('In TFrame.TrackInContent: tracker = NIL');
16 3224 --                  tracker := gNoChanges;
16 3225 --                  gotATracker := FALSE;
16 3226 --              END;
16 3227 --      {$ENDC}
16 3228 --
16 3229 --          IF gotATracker THEN
16 3230 --              BEGIN
16 3231 --                  newTracker := tracker.TrackMouse(aTrackPhase, anchorPoint,
16 3232 --                      previousPoint, theMouse, didMouseMove);
16 3233 --                  IF newTracker <> tracker THEN
16 3234 --                      BEGIN
16 3235 --                          tracker.Free;
16 3236 --                          InstallTracker(newTracker);
16 3237 --                      END
16 3238 --                  ELSE IF (newTracker <> gNoChanges) & (newTracker.fView <> view) THEN
16 3239 --                      InstallTracker(newTracker);
16 3240 --                  END;
16 3241 --0 B      END;
16 3242 --
16 3243 --0 A      BEGIN
16 3244 --          InstallTracker(theCommand);
16 3245 --
16 3246 --          SetupFocus;
16 3247 --
16 3248 --          theQDMouse := globalMouse;
16 3249 --          IF view <> NIL THEN
16 3250 --              BEGIN
16 3251 --                  GlobalToLocal(theQDMouse);
16 3252 --                  view.QDToViewPt(theQDMouse, theMouse);
16 3253 --              END
16 3254 --          ELSE
16 3255 --              PtToVpt(theQDMouse, theMouse);
16 3256 --          anchorPoint := theMouse;
16 3257 --          previousPoint := theMouse;
16 3258 --
16 3259 --          ConstrainOnce;
16 3260 --
16 3261 --          anchorPoint := theMouse;
16 3262 --          previousPoint := theMouse;      { in case Constrain changed the localPoint;
16 3263 --                                          guarantee that all 3 are the same on TrackPress }
16 3264 --
16 3265 --          TrackOnce(trackPress, TRUE);
16 3266 --          previousPoint := theMouse;      { in case TrackMouse changed nextPoint }

```

```

16 3267 --      FeedbackOnce(TRUE, TRUE);
16 3268 --
16 3269 --      movedOnce := FALSE;
16 3270 --
16 3271 --      WHILE StillDown DO
16 3272 1-          BEGIN
16 3273 --              DoFocus;
16 3274 --              GetMouse(theQDMouse);
16 3275 --              IF view <> NIL THEN
16 3276 --                  view.QDToViewPt(theQDMouse, theMouse)
16 3277 --              ELSE
16 3278 --                  PtToVpt(theQDMouse, theMouse);
16 3279 --
16 3280 --              IF NOT movedOnce THEN
16 3281 2-                  BEGIN
16 3282 --                      amtMoved := theMouse;
16 3283 --                      SubVpt(anchorPoint, amtMoved);
16 3284 --                      IF (Abs(amtMoved.h) >= hysteresis.h) OR
16 3285 --                         (Abs(amtMoved.v) >= hysteresis.v) THEN
16 3286 --                          movedOnce := TRUE;
16 3287 -2                      END;
16 3288 --
16 3289 --              IF (movedOnce | tracker.fTrackNonMovement) THEN
16 3290 2-                  BEGIN
16 3291 --                      SetVpt(delta, 0, 0);
16 3292 --
16 3293 --                      IF gotATracker THEN
16 3294 3-                          BEGIN
16 3295 --                              { ??? Problems with this:
16 3296 --                                  delta might be non-zero but scrolling can't take place
16 3297 --                                  because it is pinned at the end of the view
16 3298 --                                  also might want some slop before scrolling ??? }
16 3299 --                              IF scroller <> NIL THEN
16 3300 --                                  IF NOT PtInVRect(theMouse, autoscrollLimit) THEN
16 3301 4-                                      BEGIN
16 3302 --                                          { !!! Need to convert theMouse to scroller coordinates }
16 3303 --                                          scroller.AutoScroll(theMouse, delta);
16 3304 --                                          AddVpt(delta, theMouse);
16 3305 -4                                          END;
16 3306 --
16 3307 --                                          ConstrainOnce;
16 3308 -3                                          END;
16 3309 --
16 3310 --                                          didScroll := (delta.h <> 0) OR (delta.v <> 0);
16 3311 --                                          didMove := NOT EqualVpt(previousPoint, theMouse);
16 3312 --
16 3313 --                                          FeedbackOnce(FALSE, didMove OR didScroll);
16 3314 --
16 3315 --                                          { do the test here to avoid the method call, even though ScrollBy also checks }
16 3316 --                                          IF didScroll THEN
16 3317 3-                                              BEGIN
16 3318 --                                                  tracker.AutoScroll(delta.h, delta.v);
16 3319 --                                                  SetupFocus; { the focus changed }
16 3320 -3                                                  END;
16 3321 --
16 3322 --                                          TrackOnce(trackMove, didMove); { ??? add OR didscroll ??? }
16 3323 --
16 3324 --                                          previousPoint := theMouse;
16 3325 --                                          FeedbackOnce(TRUE, didMove OR didScroll);
16 3326 -2                                          END;
16 3327 -1                      END;
16 3328 --
16 3329 --              DoFocus;
16 3330 --
16 3331 --              IF NOT movedOnce THEN
16 3332 --                  theMouse := previousPoint { normally same as original mouse down; we don't use
16 3333 --                                                  anchorPoint in case someone has changed that -- it is more
16 3334 --                                                  likely that an app would change anchorPoint than previousPoint }
16 3335 --
16 3336 --              ELSE IF EventAvail(mUpMask, peekEvent) THEN
16 3337 1-                  BEGIN
16 3338 --                      theQDMouse := peekEvent.where;
16 3339 --                      IF view <> NIL THEN
16 3340 2-                          BEGIN
16 3341 --                              GlobalToLocal(theQDMouse);
16 3342 --                              view.QDToViewPt(theQDMouse, theMouse);
16 3343 -2                          END
16 3344 --                      ELSE
16 3345 --                          PtToVpt(theQDMouse, theMouse);
16 3346 --                          ConstrainOnce;
16 3347 -1                          END;
16 3348 --                      { ELSE we use the last known mouse position }
16 3349 --
16 3350 --                      FeedbackOnce(FALSE, TRUE);
16 3351 --                      TrackOnce(trackRelease, TRUE);
16 3352 --
16 3353 --                      TrackMouse := tracker;
16 3354 --
16 3355 -0 A      END { TApplication.TrackMouse };
16 3356 --
16 3357 --      {$S MAREs}
16 3358 --      {-----+
16 3359 --      |   UpdateAllWindows   |
16 3360 --      +-----}
16 3361 -- A      PROCEDURE TApplication.UpdateAllWindows;
16 3362 --
16 3363 --      VAR
16 3364 --          anEvent:      EventRecord;
16 3365 --

```



```
16 3366 0- A BEGIN
16 3367 --      WHILE GetEvent(updateMask, 0, NIL, anEvent) DO
16 3368 --          HandleEvent(anEvent);
16 3369 -0 A END;
16 3370 --
16 3371 --
16 3372 --      {-----+
16 3373 --      | WMgrToWindow |
16 3374 --      +-----}
16 3375 -- A FUNCTION TApplication.WMgrToWindow (aWMgrWindow: WindowPtr): TWindow;
16 3376 --
16 3377 0- A BEGIN
16 3378 --      IF (aWMgrWindow <> NIL) & (NOT IsDeskAccessory(aWMgrWindow))
16 3379 --      {SIFC qDebug}
16 3380 --          & (aWMgrWindow <> gDebugWindowPtr)
16 3381 --      {$ENDC}
16 3382 --      THEN
16 3383 --          WMgrToWindow := TWindow(GetWRefCon(aWMgrWindow))
16 3384 --      ELSE
16 3385 --          WMgrToWindow := NIL;
16 3386 -0 A END;
13 3004 --      {$I UMacApp.TDocument.p}
```

```

17 1 -- {SP}
17 2 -- { UMacApp.TDocument.p }
17 3 -- { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
17 4 --
17 5 --
17 6 -- {$S MAOpen}
17 7 -- {-----+
17 8 -- | IDocument |
17 9 -- +-----}
17 10 -- A PROCEDURE TDocument.IDocument(itsFileType, itsCreator: OSType;
17 11 -- usesDataFork, usesRsrcFork: BOOLEAN;
17 12 -- keepsDataOpen, keepsRsrcOpen: BOOLEAN);
17 13 -- VAR
17 14 -- fi: FailInfo;
17 15 --
17 16 -- {-----+
17 17 -- | HdliDocument |
17 18 -- +-----}
17 19 -- B PROCEDURE HdliDocument(error: OSErr; message: LONGINT);
17 20 -- B BEGIN
17 21 -- Free;
17 22 -- B END;
17 23 --
17 24 -- A BEGIN
17 25 -- fWindowList := NIL;
17 26 -- fViewList := NIL;
17 27 -- fPrintInfo := NIL;
17 28 -- fDocPrintHandler := NIL;
17 29 --
17 30 -- fDataRefnum := kNoFileRefnum;
17 31 -- fRsrcRefnum := kNoFileRefnum;
17 32 --
17 33 -- fTitle := '';
17 34 -- fPrintInfo := NIL;
17 35 -- fSavePrintInfo := FALSE;
17 36 -- fSharePrintInfo := TRUE;
17 37 --
17 38 -- fVolRefNum := 0;
17 39 -- fReopenAlert := TRUE;
17 40 -- fSaveExists := FALSE;
17 41 -- fCommitOnSave := TRUE;
17 42 -- fModDate := 0;
17 43 --
17 44 -- fFileType := itsFileType;
17 45 -- fCreator := itsCreator;
17 46 --
17 47 -- fUsesDataFork := usesDataFork;
17 48 -- fUsesRsrcFork := usesRsrcFork;
17 49 -- fDataOpen := keepsDataOpen;
17 50 -- fRsrcOpen := keepsRsrcOpen;
17 51 --
17 52 -- {$IFC qDebug}
17 53 -- IF fDataOpen THEN
17 54 -- IF NOT fUsesDataFork THEN
17 55 -- BEGIN
17 56 -- Writeln('In TDocument.IDocument: fDataOpen AND NOT fUsesDataFork;');
17 57 -- Writeln('In TDocument.IDocument: fUsesDataFork := TRUE;');
17 58 -- fUsesDataFork := TRUE;
17 59 -- END;
17 60 -- IF fRsrcOpen THEN
17 61 -- IF NOT fUsesRsrcFork THEN
17 62 -- BEGIN
17 63 -- Writeln('In TDocument.IDocument: fRsrcOpen AND NOT fUsesRsrcFork;');
17 64 -- Writeln('In TDocument.IDocument: fUsesRsrcFork := TRUE;');
17 65 -- fUsesRsrcFork := TRUE;
17 66 -- END;
17 67 -- {$ENDC}
17 68 --
17 69 -- IF keepsDataOpen OR keepsRsrcOpen THEN
17 70 -- fSaveInPlace := sipNever
17 71 -- ELSE
17 72 -- fSaveInPlace := sipAskUser;
17 73 --
17 74 -- fDataPerm := fsRdPerm;
17 75 -- fRsrcPerm := fsRdPerm;
17 76 --
17 77 -- IEvtHandler(qApplication);
17 78 --
17 79 -- CatchFailures(fi, HdliDocument);
17 80 --
17 81 -- fChangeCount := 0;
17 82 --
17 83 -- fWindowList := NewList;
17 84 -- {$IFC qDebug}
17 85 -- fWindowList.SetEltType('TWindow');
17 86 -- {$ENDC}
17 87 --
17 88 -- fViewList := NewList;
17 89 -- {$IFC qDebug}
17 90 -- fViewList.SetEltType('TView');
17 91 -- {$ENDC}
17 92 --
17 93 -- Success(fi);
17 94 -- A END;
17 95 --
17 96 --
17 97 -- {$S MAClose}
17 98 -- {-----+

```

```

17 99 -- | Free
17 100 -- +-----+
17 101 -- A PROCEDURE TDocument.Free; OVERRIDE;
17 102 --
17 103 0- A BEGIN
17 104 --     qApplication.DeleteDocument(SELF);
17 105 --
17 106 --     FreeFile;
17 107 --
17 108 --     IF fWindowList <> NIL THEN
17 109 --         fWindowList.FreeList;
17 110 --
17 111 --     IF fViewList <> NIL THEN
17 112 --         fViewList.FreeList;
17 113 --
17 114 --     IF fSharePrintInfo AND (fPrintInfo <> NIL) THEN
17 115 --         DisposHandle(fPrintInfo);
17 116 --
17 117 --     INHERITED Free;
17 118 -0 A END;
17 119 --
17 120 --
17 121 --     {$S MAOpen}
17 122 --     {-----+
17 123 --     | AddView
17 124 --     +-----+
17 125 -- A PROCEDURE TDocument.AddView(aView: TView);
17 126 0- A BEGIN
17 127 --     fViewList.InsertLast(aView);
17 128 -0 A END;
17 129 --
17 130 --
17 131 --     {$S MAWriteFile}
17 132 --     {-----+
17 133 --     | AboutToSave
17 134 --     +-----+
17 135 -- A PROCEDURE TDocument.AboutToSave(itsCmd: CmdNumber;
17 136 --                                     VAR newName: Str255; VAR newVolRefnum: INTEGER;
17 137 --                                     VAR makingCopy: BOOLEAN);
17 138 0- A BEGIN
17 139 -0 A END;
17 140 --
17 141 --
17 142 --     {$S MAOpen}
17 143 --     {-----+
17 144 --     | AddWindow
17 145 --     +-----+
17 146 -- A PROCEDURE TDocument.AddWindow(aWindow: TWindow);
17 147 0- A BEGIN
17 148 --     fWindowList.InsertLast(aWindow);
17 149 -0 A END;
17 150 --
17 151 --
17 152 --     {$S MAFile}
17 153 --     {-----+
17 154 --     | CheckDiskFile
17 155 --     +-----+
17 156 -- A PROCEDURE TDocument.CheckDiskFile(rsrcId, rsrcIndex: INTEGER; reverting: BOOLEAN);
17 157 --     VAR err: OSERR;
17 158 --         name: Str255;
17 159 --         s: Str255;
17 160 0- A BEGIN
17 161 --     err := DiskFileChanged(reverting); {don't care about the file type if
17 162 --                                         saving}
17 163 --
17 164 1- IF err = errFileChanged THEN
17 165 --     BEGIN
17 166 --         name := fTitle;
17 167 --         GetIndString(s, rsrcId, rsrcIndex);
17 168 --         ParamText(name, s, '', '');
17 169 --         IF MacAppAlert(phFileChanged, NIL) = cancel THEN
17 170 -1             Failure(noErr, msgCancelled);
17 171 --         END
17 172 --         {if reverting then we signal an error}
17 173 --     ELSE IF (err <> noErr) AND reverting THEN
17 174 -0 A         Failure(err, 0);
17 175 --     END;
17 176 --
17 177 --     {$S MAClose}
17 178 --     {-----+
17 179 --     | Close
17 180 --     +-----+
17 181 -- A PROCEDURE TDocument.Close;
17 182 --     {Must never be called for a document related to a view in the Clipboard.
17 183 --     Why is this???}
17 184 --
17 185 --     {-----+
17 186 --     | CloseAWindow
17 187 --     +-----+
17 188 -- B PROCEDURE CloseAWindow(aWindow: TWindow);
17 189 0- B BEGIN
17 190 --     aWindow.Close
17 191 -0 B END;
17 192 --
17 193 0- A BEGIN
17 194 --     {$IFC qDebug}
17 195 --     IF gClipWindow.fDocument = SELF THEN
17 196 --         ProgramBreak('TDocument.Close: Attempt to close clipboard document');
17 197 --     {$ENDC}

```

```

17 198 --
17 199 --     IF fChangeCount <> 0 THEN
17 200 1-         CASE PoseSaveDialog OF
17 201 --             cancel:
17 202 --                 Failure(noErr, msgCancelled);
17 203 --                 kYesButton:
17 204 --                     (Will fail if unable to save)
17 205 --                     Save(ccClose, (askForFilename:) NOT fSaveExists, kSwitchToTarget);
17 206 -1                 END;
17 207 --
17 208 --     IF gLastCommand <> NIL THEN
17 209 --         IF gLastCommand.fChangedDocument = SELF THEN
17 210 --             gApplication.CommitLastCommand;
17 211 --
17 212 --         ForAllWindowsDo(CloseAWindow);
17 213 --
17 214 --         Free;
17 215 -0 A     END;
17 216 --
17 217 --
17 218 --     {$$ MAClose}
17 219 --     {-----+
17 220 --     |   DeleteView
17 221 --     +-----}
17 222 -- A     PROCEDURE TDocument.DeleteView(viewToDelete: TView);
17 223 0- A     BEGIN
17 224 --         fViewList.Delete(viewToDelete);
17 225 -0 A     END;
17 226 --
17 227 --
17 228 --     {$$ MAClose}
17 229 --     {-----+
17 230 --     |   DeleteWindow
17 231 --     +-----}
17 232 -- A     PROCEDURE TDocument.DeleteWindow(windowToDelete: TWindow);
17 233 0- A     BEGIN
17 234 --         fWindowList.Delete(windowToDelete);
17 235 -0 A     END;
17 236 --
17 237 --
17 238 --     {$$ MAFile}
17 239 --     {-----+
17 240 --     |   DiskFileChanged
17 241 --     +-----}
17 242 -- A     FUNCTION TDocument.DiskFileChanged(checkType: BOOLEAN): OSErr;
17 243 --         VAR pb:   HParamBlockRec;
17 244 --             err:   OSErr;
17 245 0- A     BEGIN
17 246 --         IF fSaveExists THEN
17 247 1-             BEGIN
17 248 --                 err := GetFileInfo(fTitle, fVolRefnum, pb);
17 249 --                 IF err = noErr THEN
17 250 --                     IF checkType AND (pb.ioFlFndrInfo.fdType <> fFileType) THEN
17 251 --                         err := errFTypeChanged
17 252 --                         ELSE IF pb.ioFlMdDat <> fModDate THEN
17 253 --                             err := errFileChanged;
17 254 --                         DiskFileChanged := err;
17 255 -1                     END
17 256 --                 ELSE
17 257 --                     DiskFileChanged := noErr;
17 258 -0 A     END;
17 259 --
17 260 --
17 261 --     {$$ MAOpen}
17 262 --     {-----+
17 263 --     |   DoInitialState
17 264 --     +-----}
17 265 -- A     PROCEDURE TDocument.DoInitialState;
17 266 --         (Called for 'New' & 'Revert' [to blank] commands & for default open tool icon)
17 267 0- A     BEGIN
17 268 -0 A     END;
17 269 --
17 270 --
17 271 --     {$$ MANever}
17 272 --     {-----+
17 273 --     |   DoMakeViews
17 274 --     +-----}
17 275 -- A     PROCEDURE TDocument.DoMakeViews(forPrinting: BOOLEAN);
17 276 --         (* VAR aYOURView: TYOURView; *)
17 277 0- A     BEGIN
17 278 --         (*
17 279 --             NEW(aYOURView);
17 280 --             aYOURView.IYOURView(SELF, YOURExtentRect);
17 281 --             DoMakeView := aYOURView;
17 282 --         *)
17 283 -0 A     END;
17 284 --
17 285 --
17 286 --     {$$ MAREs}
17 287 --     {-----+
17 288 --     |   DoMakeWindows
17 289 --     +-----}
17 290 -- A     PROCEDURE TDocument.DoMakeWindows;
17 291 0- A     BEGIN
17 292 -0 A     END;
17 293 --
17 294 --
17 295 --     {$$ MASElCommand}
17 296 --     {-----+

```

```

17 297 -- | DoMenuCommand |
17 298 -- +-----+
17 299 -- A FUNCTION TDocument.DoMenuCommand(aCmdNumber: CmdNumber): TCommand;
17 300 --     VAR name: Str255;
17 301 --     info: HParamBlockRec;
17 302 --     err: OSErr;
17 303 --     fi: FailInfo;
17 304 --
17 305 -- {-----+
17 306 -- | HdLRevertCmd |
17 307 -- +-----+
17 308 -- B PROCEDURE HdLRevertCmd(error: OSErr; message: LONGINT);
17 309 0- B BEGIN
17 310 --     ShowReverted; {make sure screen is updated}
17 311 -0 B END;
17 312 --
17 313 0- A BEGIN
17 314 --     DoMenuCommand := gNoChanges;
17 315 --
17 316 1- CASE aCmdNumber OF
17 317 --
17 318 --         cPrFileBase..cPrFileMax:
17 319 --             IF fDocPrintHandler <> NIL THEN
17 320 --                 DoMenuCommand := fDocPrintHandler.DoMenuCommand(aCmdNumber);
17 321 --
17 322 --         cSave, cSaveAs, cSaveCopy:
17 323 2- BEGIN
17 324 --             Save(aCmdNumber,
17 325 --                 {askForFilename:} NOT fSaveExists OR (aCmdNumber <> cSave),
17 326 --                 {makingCopy:} aCmdNumber = cSaveCopy);
17 327 -2 END;
17 328 --
17 329 --         cRevert:
17 330 2- BEGIN
17 331 --             name := fName;
17 332 --             ParamText(name, '', '', '');
17 333 --             IF MacAppAlert(phRevert, NIL) = kYesButton THEN
17 334 3- BEGIN
17 335 --                 CatchFailures(fi, HdLRevertCmd);
17 336 --                 Revert;
17 337 --                 Success(fi);
17 338 --                 ShowReverted;
17 339 -3 END;
17 340 -2 END;
17 341 --
17 342 --         OTHERWISE
17 343 --             DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
17 344 -1 END; {CASE}
17 345 -0 A END;
17 346 --
17 347 --
17 348 -- {SS MAWriteFile}
17 349 -- {-----+
17 350 -- | DoNeedDiskSpace |
17 351 -- +-----+
17 352 -- A PROCEDURE TDocument.DoNeedDiskSpace(VAR dataForkBytes, rsrcForkBytes: LONGINT);
17 353 0- A BEGIN
17 354 --     IF fSavePrintInfo THEN
17 355 --         dataForkBytes := dataForkBytes + kPrintInfoSize;
17 356 --     IF fUsesRsrcFork THEN
17 357 --         rsrcForkBytes := rsrcForkBytes + kRsrcFileOverhead;
17 358 -0 A END;
17 359 --
17 360 --
17 361 -- {SS MAREadFile}
17 362 -- {-----+
17 363 -- | DoRead |
17 364 -- +-----+
17 365 -- A PROCEDURE TDocument.DoRead(aRefNum: INTEGER;
17 366 --                               rsrcExists, forPrinting: BOOLEAN);
17 367 --     VAR count: LONGINT;
17 368 --
17 369 0- A BEGIN
17 370 --     IF fSavePrintInfo THEN
17 371 1- BEGIN
17 372 --         IF fPrintInfo = NIL THEN
17 373 2- BEGIN
17 374 --             fPrintInfo := NewPermHandle(kPrintInfoSize);
17 375 --             FailNIL(fPrintInfo);
17 376 -2 END;
17 377 --
17 378 --             count := kPrintInfoSize;
17 379 --             FailOSerr(FSRead(aRefNum, count, fPrintInfo));
17 380 -1 END;
17 381 -0 A END;
17 382 --
17 383 --
17 384 -- {SS MAREs}
17 385 -- {-----+
17 386 -- | DoSetupMenus |
17 387 -- +-----+
17 388 -- A PROCEDURE TDocument.DoSetupMenus;
17 389 0- A BEGIN
17 390 --     INHERITED DoSetupMenus;
17 391 --
17 392 --     Enable(cSaveAs, TRUE);
17 393 --     Enable(cSaveCopy, TRUE);
17 394 --     IF fChangeCount <> 0 THEN
17 395 1- BEGIN

```

```

17 396 --      Enable(cSave, TRUE);
17 397 --      Enable(cRevert, TRUE);
17 398 -1      END;
17 399 --
17 400 --      IF fDocPrintHandler <> NIL THEN
17 401 --          IF NOT gTarget.HandlesPrintingCommands THEN
17 402 --              fDocPrintHandler.DoSetUpMenus;
17 403 -0 A  END;
17 404 --
17 405 --
17 406 --      {$S MAWriteFile}
17 407 --      {-----+
17 408 --      |   DoWrite
17 409 --      +-----}
17 410 -- A  PROCEDURE TDocument.DoWrite(aRefNum: INTEGER; makingCopy: BOOLEAN);
17 411 --      VAR count: LONGINT;
17 412 -0 A  BEGIN
17 413 --      IF fSavePrintInfo THEN
17 414 -1      BEGIN
17 415 --      IF fPrintInfo = NIL THEN
17 416 -2      BEGIN
17 417 --      {$IFC qDebug}
17 418 --      ProgramBreak('no print record in document') (???);
17 419 --      {$ENDC}
17 420 -2      END
17 421 --      ELSE
17 422 -2      BEGIN
17 423 --      count := kPrintInfoSize;
17 424 --      FailOSErr(FSWrite(aRefNum, count, fPrintInfo^));
17 425 -2      END;
17 426 -1      END;
17 427 -0 A  END;
17 428 --
17 429 --
17 430 --      {This is specified to work only on views currently in frames.
17 431 --      If you want all views use fViewList.Each.}
17 432 --      {$S MAREs}
17 433 --      {-----+
17 434 --      |   ForAllViewsDo
17 435 --      +-----}
17 436 -- A  PROCEDURE TDocument.ForAllViewsDo(PROCEDURE DoToView(aView: TView));
17 437 --
17 438 -0 A  BEGIN
17 439 --      IF gFinderPrinting THEN
17 440 -1      BEGIN
17 441 --      IF fDocPrintHandler <> NIL THEN
17 442 --      IF fDocPrintHandler.fView <> NIL THEN
17 443 --      DoToView(fDocPrintHandler.fView);
17 444 -1      END
17 445 --      ELSE
17 446 --      fViewList.Each(DoToView);
17 447 -0 A  END;
17 448 --
17 449 --
17 450 --      {$S MAREs}
17 451 --      {-----+
17 452 --      |   ForAllWindowsDo
17 453 --      +-----}
17 454 -- A  PROCEDURE TDocument.ForAllWindowsDo(PROCEDURE DoToWind(aWindow: TWindow));
17 455 -0 A  BEGIN
17 456 --      fWindowList.Each(DoToWind);
17 457 -0 A  END;
17 458 --
17 459 --
17 460 --      {$S MAREs}
17 461 --      {-----+
17 462 --      |   FreeData
17 463 --      +-----}
17 464 -- A  PROCEDURE TDocument.FreeData;
17 465 -0 A  BEGIN
17 466 -0 A  END;
17 467 --
17 468 --
17 469 --      {$S MAWriteFile}
17 470 --      {-----+
17 471 --      |   FreeFile
17 472 --      +-----}
17 473 -- A  PROCEDURE TDocument.FreeFile;
17 474 --      VAR err: OSErr;
17 475 -0 A  BEGIN
17 476 --      IF fDataOpen OR fRsrcOpen THEN
17 477 -1      BEGIN
17 478 --      err := CloseFile(fDataRefnum, fRsrcRefnum);
17 479 --      {$IFC qDebug}
17 480 --      IF err <> noErr THEN
17 481 --      Writeln('In TDocument.FreeFile: error from CloseFile = ', err:1);
17 482 --      {$ENDC}
17 483 -1      END;
17 484 -0 A  END;
17 485 --
17 486 --
17 487 --      {$S MAClipboard}
17 488 --      {-----+
17 489 --      |   FreeFromClipboard
17 490 --      +-----}
17 491 -- A  PROCEDURE TDocument.FreeFromClipboard;
17 492 -0 A  BEGIN
17 493 --      fWindowList.Delete(gClipWindow);
17 494 --      Free;

```

```

17 495 -0 A END;
17 496 --
17 497 --
17 498 -- ($IFC qInspector)
17 499 -- {$S MAInspector}
17 500 -- {-----+
17 501 -- | GetInspectorName |
17 502 -- +-----}
17 503 -- A PROCEDURE TDocument.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
17 504 --
17 505 0- A BEGIN
17 506 --     inspectorName := fName;
17 507 -0 A END;
17 508 -- ($ENDC)
17 509 --
17 510 --
17 511 -- {$S MAWriteFile}
17 512 -- {-----+
17 513 -- | GetSaveInfo |
17 514 -- +-----}
17 515 -- A FUNCTION TDocument.GetSaveInfo(itsCmdNumber: CmdNumber;
17 516 --                                     copyFInfo: BOOLEAN;
17 517 --                                     VAR cInfo: CInfoPRec): BOOLEAN;
17 518 --     VAR currName: Str255;
17 519 --     err: OSerr;
17 520 0- A BEGIN
17 521 --     IF fSaveExists AND copyFInfo THEN
17 522 1- BEGIN
17 523 --         currName := fName;
17 524 --         WITH cInfo DO
17 525 2- BEGIN
17 526 --             ioNamePtr := @currName;
17 527 --             ioVRefnum := fVolRefnum;
17 528 --             ioFVersNum := 0;
17 529 --             ioFDirIndex := 0;
17 530 --             ioDirID := 0;
17 531 -2 END;
17 532 --             IF gConfiguration.hashFS THEN
17 533 2- BEGIN
17 534 --                 err := FillInDirID(@cInfo);
17 535 --                 IF err = noErr THEN
17 536 --                     err := PBGetCatInfo(@cInfo, FALSE);
17 537 -2 END
17 538 --             ELSE
17 539 --                 err := PBGetFInfo(@cInfo, FALSE);
17 540 --
17 541 --             cInfo.ioNamePtr := NIL; {since ptr is invalid when we exit}
17 542 --
17 543 --             {set the type and creator in case it has changed;
17 544 --              the file might be on a file server and someone else
17 545 --              could have changed the document}
17 546 --             cInfo.ioFlFndrInfo.fdCreator := fCreator;
17 547 --             cInfo.ioFlFndrInfo.fdType := fFileType;
17 548 -1 END
17 549 --         ELSE
17 550 --             err := fnfErr; {fake error}
17 551 --
17 552 --         IF err = noErr THEN
17 553 --             GetSaveInfo := TRUE
17 554 --         ELSE
17 555 1- BEGIN
17 556 --             WITH cInfo.ioFlFndrInfo DO
17 557 2- BEGIN
17 558 --                 fdCreator := fCreator;
17 559 --                 fdType := fFileType;
17 560 -2 END;
17 561 --
17 562 --             GetSaveInfo := FALSE
17 563 -1 END;
17 564 -0 A END;
17 565 --
17 566 --
17 567 -- {$S MAFile}
17 568 -- {-----+
17 569 -- | GetTempName |
17 570 -- +-----}
17 571 -- A PROCEDURE TDocument.GetTempName(VAR filename: Str255);
17 572 --
17 573 -- CONST
17 574 --     maxName = 31; {maximum name size to generate}
17 575 --     maxNumber = 10; {maximum # digits of the random number}
17 576 --     maxPrefix = maxName - maxNumber;
17 577 --
17 578 -- VAR
17 579 --     s: Str255;
17 580 --     apRefnum: INTEGER;
17 581 --     apParam: Handle;
17 582 --     time: LONGINT;
17 583 --
17 584 0- A BEGIN
17 585 --     {If the document is untitled, use the application name.}
17 586 --     IF fName = '' THEN
17 587 --         GetAppParms(filename, apRefnum, apParam)
17 588 --     ELSE
17 589 --         filename := fName;
17 590 --
17 591 --     IF Length(filename) > maxPrefix THEN
17 592 --         filename := Copy(filename, 1, maxPrefix);
17 593 --

```

```

17 594 --      (append a pseudo-random number)
17 595 --      GetDateTime(time);
17 596 --      NumToString(ABS(BXOR(time, BRotR(TickCount, 16))), s);
17 597 --      filename := CONCAT(filename, s);
17 598 -0 A  END;
17 599 --
17 600 --
17 601 --      {$S MAREs}
17 602 --      {-----+
17 603 --      |   HandlesPrintingCommands   |
17 604 --      +-----+}
17 605 -- A  FUNCTION TDocument.HandlesPrintingCommands: BOOLEAN; OVERRIDE;
17 606 0- A  BEGIN
17 607 --      HandlesPrintingCommands := FALSE;
17 608 -0 A  END;
17 609 --
17 610 --
17 611 --      {$S MAWriteFile}
17 612 --      {-----+
17 613 --      |   MakeNewCopy               |
17 614 --      +-----+}
17 615 -- A  PROCEDURE TDocument.MakeNewCopy(makingCopy: BOOLEAN;
17 616 --      validFInfo: BOOLEAN;
17 617 --      VAR cInfo: CInfoPbRec);
17 618 --      (This routine changes the following fields of cInfo:
17 619 --
17 620 --      The ioNamePtr, and ioVRefnum fields must be set to indicate
17 621 --      the desired file; this routine sets ioDirID to 0.
17 622 --
17 623 --      IF setFInfo is TRUE then all the fields of cInfo
17 624 --      should be set up. Otherwise only the file type and creator
17 625 --      need to be set up.)
17 626 --
17 627 --      VAR err:      OSErr;
17 628 --      dataRefnum:  INTEGER;
17 629 --      rsrcRefnum:  INTEGER;
17 630 --      oldVRefnum:  INTEGER;
17 631 --      fi:          FailInfo;
17 632 --
17 633 --      {-----+
17 634 --      |   HdLMkNewCopy               |
17 635 --      +-----+}
17 636 -- B  PROCEDURE HdLMkNewCopy(error: OSErr; message: LONGINT);
17 637 --      VAR err:      OSErr;
17 638 0- B  BEGIN
17 639 --      err := CloseFile(dataRefnum, rsrcRefnum);
17 640 --      ($IFC qDebug)
17 641 --      IF err <> noErr THEN
17 642 --          Writeln('In HdLMkNewCopy: error from CloseFile is ', err: 1);
17 643 --      ($ENDC)
17 644 --
17 645 --      err := DeleteFile(cInfo.ioNamePtr, cInfo.ioVRefnum);
17 646 --      ($IFC qDebug)
17 647 --      IF err <> noErr THEN
17 648 --          IF err <> fnfErr THEN
17 649 --              Writeln('In HdLMkNewCopy: error from DeleteFile is ', err: 1);
17 650 --          ($ENDC)
17 651 -0 B  END;
17 652 --
17 653 0- A  BEGIN
17 654 --      IF fusesDataFork OR fUsesRsrcFork THEN
17 655 1-      BEGIN
17 656 --          cInfo.ioDirID := 0;
17 657 --
17 658 --          (Initialize these in case we fail before the call to OpenAFile.)
17 659 --          dataRefnum := kNoFileRefnum;
17 660 --          rsrcRefnum := kNoFileRefnum;
17 661 --
17 662 --          (Create the file with the desired creator/type,
17 663 --          in case we are not going to set the file info
17 664 --          below.)
17 665 --          FailOSErr(Create(cInfo.ioNamePtr, cInfo.ioVRefnum,
17 666 --              cInfo.ioFlFndrInfo.fdCreator,
17 667 --              cInfo.ioFlFndrInfo.fdType));
17 668 --
17 669 --          CatchFailures(fi, HdLMkNewCopy);
17 670 --
17 671 --          IF fUsesRsrcFork THEN
17 672 2-      BEGIN
17 673 --          FailOSErr(GetVol(NIL, oldVRefnum));
17 674 --          FailOSErr(SetVol(NIL, cInfo.ioVRefnum));
17 675 --
17 676 --          CreateResFile(cInfo.ioNamePtr);
17 677 --
17 678 --          (Do this here to ensure that the current volume is
17 679 --          reset, even if we fail.)
17 680 --          FailOSErr(SetVol(NIL, oldVRefnum));
17 681 --
17 682 --          FailResError; (this checks the call to CreateResFile)
17 683 -2  END;
17 684 --
17 685 --      IF validFInfo THEN
17 686 --          IF NOT makingCopy THEN
17 687 2-      BEGIN
17 688 --          IF gConfiguration.hashFS THEN
17 689 --              err := PBSetCatInfo(@cInfo, FALSE) (??? PMSP ???)
17 690 --          ELSE
17 691 3-      BEGIN
17 692 --          (NOTE: We can use the cInfo for this call since the

```



```

17 693 --                fields required by PBSetInfo are a subset of
17 694 --                those in the CInfoPBRec; except we set the
17 695 --                ioFVersNum field (just in case))
17 696 --                cInfo.ioFVersNum := 0;
17 697 --                err := PBSetFInfo(@cInfo, FALSE);
17 698 -3                END;
17 699 --                FailOSErr(err);
17 700 -2                END;
17 701 --
17 702 --                FailOSErr(OpenAFile(cInfo.ioNamePtr^,      cInfo.ioVRefnum,
17 703 --                                fUsesDataFork,          fUsesRsrcFork,
17 704 --                                fsRdWrPerm,              fsRdWrPerm,
17 705 --                                dataRefnum,              rsrcRefnum));
17 706 --
17 707 --                DoWrite(dataRefnum, makingCopy);
17 708 --
17 709 --                Success(fi);
17 710 --
17 711 --                FailOSErr(CloseFile(dataRefnum, rsrcRefnum));
17 712 -1                END;
17 713 -0 A    END;
17 714 --
17 715 --
17 716 --                {$$ MAFFile}
17 717 --                {-----+
17 718 --                |   OpenAFile
17 719 --                +-----}
17 720 -- A    FUNCTION TDocument.OpenAFile(name: Str255; volRefnum: INTEGER;
17 721 --                openRsrc: BOOLEAN; dataPerm, rsrcPerm: INTEGER;
17 722 --                VAR dataRefnum, rsrcRefnum: INTEGER): OSErr;
17 723 --
17 724 0- A    BEGIN
17 725 --                OpenAFile := OpenFile(name, volRefnum, openData, openRsrc, dataPerm, rsrcPerm,
17 726 --                                dataRefnum, rsrcRefnum);
17 727 -0 A    END;
17 728 --
17 729 --
17 730 --                {$$ MAOpen}
17 731 --                {-----+
17 732 --                |   OpenAgain
17 733 --                +-----}
17 734 -- A    PROCEDURE TDocument.OpenAgain(itsCmdNumber: CmdNumber; openingDoc: TDocument);
17 735 --                VAR window: TWindow;
17 736 --                s:      Str255;
17 737 0- A    BEGIN
17 738 --                s := fTitle;          { because ParamText allocates memory }
17 739 --                ParamText(s, '', '', '');
17 740 --
17 741 --                IF fReopenAlert THEN
17 742 --                    StdAlert(phReopenDoc);
17 743 --
17 744 --                window := TWindow(fWindowList.First);
17 745 --                window.Select;
17 746 --                Failure(0, 0);
17 747 -0 A    END;
17 748 --
17 749 --
17 750 --                {$$ MAClose}
17 751 --                {-----+
17 752 --                |   PoseSaveDialog
17 753 --                +-----}
17 754 -- A    FUNCTION TDocument.PoseSaveDialog: INTEGER;
17 755 --
17 756 --                VAR
17 757 --                    idx:      INTEGER;
17 758 --                    name:     Str255;
17 759 --                    reason:   Str255;
17 760 --
17 761 0- A    BEGIN
17 762 --                IF fChangeCount <> 0 THEN
17 763 1-    BEGIN
17 764 --                    IF gAppDone THEN
17 765 --                        idx := bzQuitting
17 766 --                    ELSE
17 767 --                        idx := bzClosing;
17 768 --
17 769 --                    GetIndString(reason, kIDBuzzString, idx);
17 770 --                    name := fTitle;          { ParamText can compact heap }
17 771 --                    ParamText(name, reason, '', '');
17 772 --
17 773 --                    PoseSaveDialog := MacAppAlert(phSaveChanges, NIL);
17 774 -1                END
17 775 --                ELSE
17 776 --                    PoseSaveDialog := kNoButton;
17 777 -0 A    END (TDocument.PoseSaveDialog);
17 778 --
17 779 --                {$$ MAFinder}
17 780 --                {-----+
17 781 --                |   Print
17 782 --                +-----}
17 783 --
17 784 -- A    FUNCTION TDocument.Print(itsCmdNumber: CmdNumber): BOOLEAN;
17 785 --                {??? Only relevant for print from Finder at the present, so aCommand will always be
17 786 --                gNoChanges, and hence need not have anything done to it. May change ???}
17 787 --                VAR proceed:   BOOLEAN;
17 788 --                aCommand:   TCommand;
17 789 0- A    BEGIN
17 790 --                aCommand := fDocPrintHandler.Print(itsCmdNumber, proceed);
17 791 --                Print := proceed;

```

```

17 792 -0 A      END;
17 793 --      {$ENDC}
17 794 --
17 795 --      {$S MReadFile}
17 796 --      {-----+
17 797 --      |   ReadFromFile
17 798 --      +-----+}
17 799 -- A  PROCEDURE TDocument.ReadFromFile(VAR anAppFile: AppFile; forPrinting: BOOLEAN);
17 800 --      VAR fi:      FailInfo;
17 801 --      dataRefnum:   INTEGER;
17 802 --      rsrcRefnum:   INTEGER;
17 803 --      isRevert:     BOOLEAN;
17 804 --      shouldOpenData: BOOLEAN;
17 805 --      shouldOpenRsrc: BOOLEAN;
17 806 --
17 807 --      {-----+
17 808 --      |   Hd1Read
17 809 --      +-----+}
17 810 -- B  PROCEDURE Hd1Read(error: INTEGER; message: LONGINT);
17 811 --      VAR err:      INTEGER;
17 812 -- B  BEGIN
17 813 --      err := CloseFile(dataRefnum, rsrcRefnum);
17 814 --      ($IFC qDebug)
17 815 --      IF err <> noErr THEN
17 816 --          WriteLn('In Hd1Open: error from CloseFile is ', err:1);
17 817 --      {$ENDC}
17 818 --
17 819 --      {caller should set the message as appropriate}
17 820 -- B  END;
17 821 --
17 822 -- A  BEGIN
17 823 --      isRevert := (anAppFile.fName = ''); {signal to re-read file}
17 824 --      IF isRevert THEN
17 825 --          BEGIN
17 826 --              anAppFile.fName := fName;
17 827 --              anAppFile.vRefnum := fVolRefnum;
17 828 --          END
17 829 --      ELSE
17 830 --          BEGIN
17 831 --              fName := anAppFile.fName;
17 832 --              fVolRefnum := anAppFile.vRefnum;
17 833 --          END;
17 834 --
17 835 --      {Don't attempt to open the fork(s) again if they're already open.
17 836 --      We do all this rather than close and reopen so that we need not
17 837 --      search the directory again, an expensive operation on large MFS
17 838 --      disks.}
17 839 --      shouldOpenData := fUsesDataFork AND NOT (fDataOpen AND isRevert);
17 840 --      shouldOpenRsrc := fUsesRsrcFork AND NOT (fRsrcOpen AND isRevert);
17 841 --
17 842 --      {Make sure CloseFile operates properly if OpenAFile fails.}
17 843 --      dataRefnum := kNoFileRefnum;
17 844 --      rsrcRefnum := kNoFileRefnum;
17 845 --
17 846 --      CatchFailures(fi, Hd1Read);
17 847 --
17 848 --      FailOnError(OpenAFile(anAppFile.fName, anAppFile.vRefnum,
17 849 --          shouldOpenData, shouldOpenRsrc,
17 850 --          fDataPerm, fRsrcPerm,
17 851 --          dataRefnum, rsrcRefnum));
17 852 --
17 853 --      fSaveExists := TRUE;
17 854 --
17 855 --      {If the file is already open, use the refnums we already have.
17 856 --      Make sure that the document's data fork is positioned at TOF.
17 857 --      Make sure that the document's resource file is on top.}
17 858 --      IF fDataOpen AND NOT shouldOpenData THEN
17 859 --          BEGIN
17 860 --              dataRefnum := fDataRefnum;
17 861 --              FailOnError(SetFPos(dataRefnum, fsFromStart, 0));
17 862 --          END;
17 863 --      IF fRsrcOpen AND NOT shouldOpenRsrc THEN
17 864 --          BEGIN
17 865 --              rsrcRefnum := fRsrcRefnum;
17 866 --              UseResFile(rsrcRefnum);
17 867 --          END;
17 868 --
17 869 --      DoRead(dataRefnum, {rsrcExists:} rsrcRefnum <> kNoFileRefnum, forPrinting);
17 870 --
17 871 --      fChangeCount := 0;
17 872 --
17 873 --      Success(fi);
17 874 --
17 875 --      IF fDataOpen THEN
17 876 --          BEGIN
17 877 --              fDataRefnum := dataRefnum;      {save valid refnum}
17 878 --              dataRefnum := kNoFileRefnum;    {make sure CloseFile doesn't close it}
17 879 --          END
17 880 --      ELSE
17 881 --          fDataRefnum := kNoFileRefnum;      {save the no file refnum}
17 882 --
17 883 --      IF fRsrcOpen THEN
17 884 --          BEGIN
17 885 --              fRsrcRefnum := rsrcRefnum;      {save valid refnum}
17 886 --              rsrcRefnum := kNoFileRefnum;    {make sure CloseFile doesn't close it}
17 887 --          END
17 888 --      ELSE
17 889 --          fRsrcRefnum := kNoFileRefnum;      {save the no file refnum}
17 890 --

```

```

17 891 --      FailOnError(CloseFile(dataRefnum, rsrcRefnum));
17 892 --
17 893 --      fModDate := FileModDate(anAppFile.fName, anAppFile.vRefnum);
17 894 -0 A  END;
17 895 --
17 896 --
17 897 --      {$S MAWriteFile}
17 898 --      {-----+
17 899 --      | RequestFileName
17 900 --      +-----}
17 901 -- A  PROCEDURE TDocument.RequestFileName (itsCmdNumber: CmdNumber;
17 902 --                                         makingCopy: BOOLEAN;
17 903 --                                         VAR filename: Str255;
17 904 --                                         VAR volRefNum: INTEGER);
17 905 --
17 906 --      VAR
17 907 --          reply:          SFReply;
17 908 --          dlgID:          INTEGER;
17 909 --          prompt:         Str255;
17 910 --          dlgLoc:         Point;
17 911 --          dlgHook:        ProcPtr;
17 912 --          filterProc:     ProcPtr;
17 913 --          otherDoc:       TDocument;
17 914 --          err:            OSErr;
17 915 --
17 916 0- A  BEGIN
17 917 --          filename := fTitle;
17 918 --          SFPutParams(itsCmdNumber, dlgID, dlgLoc, filename, prompt, dlgHook, filterProc);
17 919 --
17 920 --      {$IFC qDebug}
17 921 --          gRsrcCheck := 0; {force immediate check}
17 922 --      {$ENDC}
17 923 --
17 924 --      {Update all the windows to avoid a bug in Standard File in which
17 925 --       you can't mount a disk correctly when window updates are pending.}
17 926 --      gApplication.UpdateAllWindows;
17 927 --
17 928 --      SFPPutFile(dlgLoc, prompt, filename, dlgHook, reply, dlgID, filterProc);
17 929 --
17 930 --      IF reply.good THEN
17 931 1-  BEGIN
17 932 --          filename := reply.fName;
17 933 --          volRefnum := reply.vRefnum;
17 934 --
17 935 --          {See if there is an open document with the same name. If there
17 936 --           is, tell it we're trying to save it again, which will
17 937 --           ordinarily result in failure.}
17 938 --          otherDoc := gApplication.AlreadyOpen(filename, volRefnum);
17 939 --          IF otherDoc <> NIL THEN
17 940 --              otherDoc.SaveAgain(itsCmdNumber, makingCopy, SELF);
17 941 --
17 942 --          {User has already confirmed deleting target in this case,
17 943 --           so trash file and get maximum disk space.}
17 944 --          err := DeleteFile(@filename, volRefnum);
17 945 --          IF (err <> noErr) AND (err <> fnfErr) THEN
17 946 --              Failure(err, 0);
17 947 -1  END
17 948 --      ELSE
17 949 --          Failure(noErr, msgCancelled); {user cancelled}
17 950 -0 A  END (TDocument.RequestFileName);
17 951 --
17 952 --
17 953 --      {$S MAREadFile}
17 954 --      {-----+
17 955 --      | Revert
17 956 --      +-----}
17 957 -- A  PROCEDURE TDocument.Revert;
17 958 --      VAR anAppFile: AppFile;
17 959 --          fi:         FailInfo;
17 960 --
17 961 --      {-----+
17 962 --      | Hd1Revert
17 963 --      +-----}
17 964 -- B  PROCEDURE Hd1Revert(error: OSErr; message: LONGINT);
17 965 0- B  BEGIN
17 966 --      IF error = fnfErr THEN
17 967 --          error := errRevertFNF;
17 968 --      IF message = 0 THEN
17 969 --          gErrorParm3 := fTitle;
17 970 --          FailNewMessage(error, message, msgRevertFailed);
17 971 -0 B  END;
17 972 --
17 973 --      {-----+
17 974 --      | ResetPrintHandler
17 975 --      +-----}
17 976 -- B  PROCEDURE ResetPrintHandler(view: TView);
17 977 0- B  BEGIN
17 978 --      IF view.fPrintHandler <> NIL THEN
17 979 --          view.fPrintHandler.Reset;
17 980 -0 B  END;
17 981 --
17 982 0- A  BEGIN
17 983 --      CatchFailures(fi, Hd1Revert);
17 984 --
17 985 --      CheckDiskFile(KIDBuzzString, bzRevertAnyways, (reverting:}TRUE);
17 986 --
17 987 --      IF gLastCommand <> NIL THEN
17 988 --          IF gLastCommand.fChangedDocument = SELF THEN
17 989 --              gApplication.CommitLastCommand;

```

```

17 990 --
17 991 --      FreeData;
17 992 --
17 993 --      IF fSaveExists THEN
17 994 1-          BEGIN
17 995 --              anAppFile.fName := ''; (signal that we are reverting)
17 996 --              ReadFromFile(anAppFile, kForDisplay);
17 997 -1          END
17 998 --      ELSE
17 999 1-          BEGIN
17 1000 --              fViewList.Each(ResetPrintHandler);
17 1001 --              DoInitialState;
17 1002 -1          END;
17 1003 --
17 1004 --      fChangeCount := 0;
17 1005 --
17 1006 --      Success(fi);
17 1007 -0 A  END;
17 1008 --
17 1009 --
17 1010 --      (Notes:
17 1011 --          Parameter combinations:
17 1012 --              ask?      copy?
17 1013 --              T          F      Save As... or Save of Untitled doc
17 1014 --              T          T      Save a Copy In...
17 1015 --              F          F      Save of a titled doc
17 1016 --
17 1017 --          Caller is responsible for passing askForFilename = TRUE
17 1018 --          if the document is currently untitled.
17 1019 --
17 1020 --          If askForFilename is FALSE, we force switchToTarget to TRUE.
17 1021 --
17 1022 --          We call SaveInPlace only if there is insufficient disk space
17 1023 --          to save a temporary copy and we are allowed to save in
17 1024 --          place (according to the fSaveInPlace field).
17 1025 --
17 1026 --          If askForFilename is TRUE, SaveInPlace should not be called
17 1027 --          since we delete the target immediately. (If there is not
17 1028 --          enough room to save via a temporary file, we would not
17 1029 --          be able to save in place either.)
17 1030 --      )
17 1031 --
17 1032 --      ($$ MAWriteFile)
17 1033 --      {-----+
17 1034 --      | Save |
17 1035 --      +-----}
17 1036 -- A  PROCEDURE TDocument.Save(itsCmdNumber: CmdNumber;
17 1037 --                          askForFilename, makingCopy: BOOLEAN);
17 1038 --
17 1039 --      VAR
17 1040 --          name:          Str255;
17 1041 --          volRefnum:      INTEGER;
17 1042 --
17 1043 --          hPB:            HParamBlockRec;
17 1044 --
17 1045 --          dataBytes:      LONGINT;
17 1046 --          rsrcBytes:      LONGINT;
17 1047 --          neededBlks:     LONGINT;
17 1048 --          usedBlks:       LONGINT;
17 1049 --          freeBlks:       LONGINT;
17 1050 --          blkSize:        LONGINT;
17 1051 --
17 1052 --          copyFInfo:      BOOLEAN;
17 1053 --          canSaveInPlace: BOOLEAN;
17 1054 --          oldFlag:        BOOLEAN;
17 1055 --          fi:             FailInfo;
17 1056 --          err:            OSErr;
17 1057 --
17 1058 --          otherDoc:       TDocument;
17 1059 --
17 1060 --      {-----+
17 1061 --      | Hd1Save |
17 1062 --      +-----}
17 1063 -- B  PROCEDURE Hd1Save(error: INTEGER; message: LONGINT);
17 1064 --      VAR newMsg: LONGINT;
17 1065 0- B  BEGIN
17 1066 --          err := FlushVol(NIL, volRefnum);
17 1067 --
17 1068 --          IF message = 0 THEN
17 1069 --              gErrorParm3 := name;
17 1070 --
17 1071 --          IF NOT askForFilename THEN
17 1072 --              newMsg := msgSaveFailed
17 1073 --          ELSE IF makingCopy THEN
17 1074 --              newMsg := msgSaveCopyFailed
17 1075 --          ELSE
17 1076 --              newMsg := msgSaveAsFailed;
17 1077 --
17 1078 --          FailNewMessage(error, message, newMsg);
17 1079 -0 B  END;
17 1080 --
17 1081 0- A  BEGIN
17 1082 --      CatchFailures(fi, Hd1Save);
17 1083 --
17 1084 --      {Step 1: Get the target of the save}
17 1085 --      {Caller should set askForFilename if this is an Untitled document}
17 1086 --      IF askForFilename THEN
17 1087 --          RequestFileName(itsCmdNumber, makingCopy, name, volRefnum)
17 1088 --      ELSE

```

```

17 1089 1-      BEGIN
17 1090 --      name := fName;
17 1091 --      volRefnum := fVolRefnum;
17 1092 -1      END;
17 1093 --
17 1094 --
17 1095 --      (Step 2: Decide whether to save with a temporary file or in place,
17 1096 --      and call appropriate method (SaveViaTemp or SaveInPlace).)
17 1097 --
17 1098 --      copyFInfo := NOT (askForFilename OR makingCopy);
17 1099 --
17 1100 --      IF copyFInfo THEN
17 1101 --          CheckDiskFile(kIDBuzzString, bzSaveAnyways, (reverting:}FALSE);
17 1102 --
17 1103 --      AboutToSave(itsCmdNumber, name, volRefnum, makingCopy);
17 1104 --
17 1105 --      IF fCommitOnSave OR NOT makingCopy THEN
17 1106 --          IF gLastCommand <> NIL THEN
17 1107 --              IF gLastCommand.fChangedDocument = SELF THEN
17 1108 --                  gApplication.CommitLastCommand;
17 1109 --
17 1110 --          (Get information about the volume saving to)
17 1111 --      WITH hPB DO
17 1112 1-          BEGIN
17 1113 --              ioNamePtr := NIL;
17 1114 --              ioVRefnum := volRefnum;
17 1115 --              ioVolIndex := 0;
17 1116 -1          END;
17 1117 --      FailOSErr(PBHGetVInfo(@hPB, FALSE));
17 1118 --
17 1119 --      (on HFS ioVFrBlk is an unsigned INTEGER; on MFS it is
17 1120 --      limited to a positive signed INTEGER)
17 1121 --      freeBlks := BAND(hPB.ioVFrBlk, $0000FFFF) - 1; {-1 for some slop -- don't try to fill up the disk completely)
17 1122 --
17 1123 --      (compute size needed to save document)
17 1124 --      blkSize := hPB.ioValBlkSize;
17 1125 --
17 1126 --      dataBytes := 0;
17 1127 --      rsrcBytes := 0;
17 1128 --      DoNeedDiskSpace(dataBytes, rsrcBytes);
17 1129 --      neededBlks := NumBlocks(rsrcBytes, blkSize) + NumBlocks(dataBytes, blkSize);
17 1130 --
17 1131 --      IF freeBlks >= neededBlks THEN
17 1132 --          (enough disk space to create a second copy of document)
17 1133 --          SaveViaTemp(itsCmdNumber, makingCopy, copyFInfo, name, volRefnum)
17 1134 --      ELSE
17 1135 1-          BEGIN      (cannot make a duplicate of document)
17 1136 --              (Check to see if we can save the file in place.)
17 1137 --
17 1138 --              canSaveInPlace := FALSE;      (default value)
17 1139 --
17 1140 --              IF fSaveInPlace <> sipNever THEN
17 1141 2-                  BEGIN
17 1142 --                      (See if target exists, if the disk space it uses is
17 1143 --                      enough to allow us to save the file after deleting it.)
17 1144 --                      err := GetFileInfo(name, volRefnum, hPB);
17 1145 --
17 1146 --                      IF err = noErr THEN
17 1147 3-                          BEGIN
17 1148 --                              (compute # block used by target)
17 1149 --                              usedBlks := NumBlocks(hPB.ioFlRPyLen, blkSize) +
17 1150 --                                  NumBlocks(hPB.ioFlPyLen, blkSize);
17 1151 --
17 1152 --                              IF neededBlks <= usedBlks + freeBlks THEN
17 1153 --                                  (we could save if target is deleted first)
17 1154 4-                                  BEGIN
17 1155 --                                      IF fSaveInPlace = sipAskUser THEN
17 1156 5-                                          BEGIN
17 1157 --                                              ParamText(name, '', '', '');
17 1158 --                                              IF MacAppAlert(phPurgeOld, NIL) = kYesButton THEN
17 1159 --                                                  canSaveInPlace := TRUE
17 1160 --                                              ELSE
17 1161 --                                                  Failure(noErr, msgCancelled);
17 1162 -5                                          END
17 1163 --                                              ELSE      (we know fSaveInPlace <> sipNever;
17 1164 --                                                  it must be sipAlways)
17 1165 --                                                  canSaveInPlace := TRUE;
17 1166 -4                                          END;
17 1167 -3                                          END
17 1168 --                                  ELSE IF err <> fnfErr THEN
17 1169 --                                      Failure(err, 0);
17 1170 -2                                  END;
17 1171 --
17 1172 --                                  IF canSaveInPlace THEN
17 1173 --                                      SaveInPlace(itsCmdNumber, makingCopy, copyFInfo, name, volRefnum)
17 1174 --                                  ELSE
17 1175 --                                      Failure(dskFulErr, 0);
17 1176 -1                                  END;
17 1177 --
17 1178 --                                  Success(fi);      (??? Put later in proc ???)
17 1179 --
17 1180 --          ($IFC qDebug)
17 1181 --          err := GetFileInfo(name, volRefnum, hPB);
17 1182 --          IF err = noErr THEN
17 1183 1-              BEGIN
17 1184 --                  usedBlks := NumBlocks(hPB.ioFlRPyLen, blkSize) + NumBlocks(hPB.ioFlPyLen, blkSize);
17 1185 --                  IF usedBlks <> neededBlks THEN
17 1186 2-                      BEGIN
17 1187 --                          Writeln('In TDocument.Save: DoNeedDiskSpace estimated disk space incorrectly.');
```

```

17 1188 --          WriteIn('estimated # disk blocks = ', neededBlks:1);
17 1189 --          WriteIn(' actual # disk blocks = ', usedBlks:1);
17 1190 -2          END;
17 1191 -1          END;
17 1192 --      {ENDC}
17 1193 --
17 1194 --      {Step 3: Tell the document that the save was successful.}
17 1195 --      IF NOT makingCopy THEN
17 1196 --          SavedOn(name, volRefnum);
17 1197 --
17 1198 --          err := FlushVol(NIL, volRefnum);
17 1199 -0 A      END;
17 1200 --
17 1201 --
17 1202 --      {$S MAWriteFile}
17 1203 --      {-----+
17 1204 --      |   SaveAgain   |
17 1205 --      +-----+}
17 1206 -- A      PROCEDURE TDocument.SaveAgain(itsCmdNumber: CmdNumber; makingCopy: BOOLEAN;
17 1207 --          savingDoc: TDocument);
17 1208 0- A      BEGIN
17 1209 --          {Don't save the file if another one of the same name is already open.}
17 1210 --          IF savingDoc <> SELF THEN
17 1211 --              Failure(errSaveAgain, 0);
17 1212 -0 A      END;
17 1213 --
17 1214 --
17 1215 --      {$S MAWriteFile}
17 1216 --      {-----+
17 1217 --      |   SavedOn     |
17 1218 --      +-----+}
17 1219 -- A      PROCEDURE TDocument.SavedOn(VAR fileName: Str255; volRefnum: INTEGER);
17 1220 --      VAR
17 1221 --          dataRefnum: INTEGER;
17 1222 --          rsrcRefnum: INTEGER;
17 1223 --
17 1224 0- A      BEGIN
17 1225 --          fChangeCount := 0;
17 1226 --          fSaveExists := TRUE;
17 1227 --
17 1228 --          IF fTitle <> fileName THEN
17 1229 --              SetTitle(fileName);
17 1230 --          fVolRefnum := volRefnum;
17 1231 --
17 1232 --          fModDate := FileModDate(fileName, volRefnum);
17 1233 --
17 1234 --          FailOSErr(OpenAFile(fileName, volRefnum,
17 1235 --              fDataOpen, fRsrcOpen,
17 1236 --              fDataPerm, fRsrcPerm,
17 1237 --              dataRefnum, rsrcRefnum));
17 1238 --          fDataRefnum := dataRefnum;
17 1239 --          fRsrcRefnum := rsrcRefnum;
17 1240 -0 A      END;
17 1241 --
17 1242 --
17 1243 --      {$S MAWriteFile}
17 1244 --      {-----+
17 1245 --      |   SaveInPlace  |
17 1246 --      +-----+}
17 1247 -- A      PROCEDURE TDocument.SaveInPlace(itsCmdNumber: CmdNumber;
17 1248 --          makingCopy, copyFInfo: BOOLEAN;
17 1249 --          VAR fileName: Str255; volRefnum: INTEGER);
17 1250 --          {fileName is VAR only to avoid copying}
17 1251 --      VAR cInfo: CInfoPBRec;
17 1252 --          validInfo: BOOLEAN;
17 1253 --          err: OSErr;
17 1254 0- A      BEGIN
17 1255 --          IF NOT (fDataOpen OR fRsrcOpen) THEN
17 1256 1-          BEGIN
17 1257 --              validInfo := GetSaveInfo(itsCmdNumber, copyFInfo, cInfo);
17 1258 --
17 1259 --              {Tell document that the file is going away.}
17 1260 --              FreeFile;
17 1261 --
17 1262 --              {Delete the current file.}
17 1263 --              err := DeleteFile(@fileName, volRefnum);
17 1264 --              IF err <> noErr THEN
17 1265 --                  IF err <> fnfErr THEN
17 1266 --                      Failure(err, 0);
17 1267 --
17 1268 --                      {Save a new copy.}
17 1269 --                      cInfo.ioNamePtr := @fileName;
17 1270 --                      cInfo.ioVRefnum := volRefnum;
17 1271 --
17 1272 --                      MakeNewCopy(makingCopy, validInfo, cInfo);
17 1273 -1          END
17 1274 --          ELSE
17 1275 1-          BEGIN
17 1276 --              {$IFC qDebug}
17 1277 --                  ProgramBreak('You must override TDocument.SaveInPlace for a disk-based document.');
```

```

17 1287 -- A  PROCEDURE TDocument.SaveViaTemp(itsCmdNumber: CmdNumber;
17 1288 --      makingCopy, copyFInfo: BOOLEAN;
17 1289 --      VAR fileName: Str255; volRefnum: INTEGER);
17 1290 --      (fileName is VAR only to avoid copying)
17 1291 --      VAR cInfo:      CInfoPRec;
17 1292 --      validInfo:      BOOLEAN;
17 1293 --      tmpName:        Str255;
17 1294 --      fi:             FailInfo;
17 1295 --      err:            OSErr;
17 1296 --
17 1297 --      {-----+
17 1298 --      |  Hd1SvTemp      |
17 1299 --      +-----+}
17 1300 -- B  PROCEDURE Hd1SvTemp(error: OSErr; message: LONGINT);
17 1301 --      VAR err:      OSErr;
17 1302 -- 0- B  BEGIN
17 1303 --      err := DeleteFile(@tmpName, volRefnum);
17 1304 --      {IFC qDebug}
17 1305 --      IF err <> noErr THEN
17 1306 --          IF err <> fnfErr THEN
17 1307 --              Writeln('In Hd1SvTemp: error from DeleteFile is ', err:1);
17 1308 --          {$ENDC}
17 1309 -- 0 B  END;
17 1310 --
17 1311 --      {-----+
17 1312 --      |  Hd1SaveFailed  |
17 1313 --      +-----+}
17 1314 -- B  PROCEDURE Hd1SaveFailed(error: OSErr; message: LONGINT);
17 1315 --      VAR
17 1316 --          dataRefnum: INTEGER;
17 1317 --          rsrcRefnum: INTEGER;
17 1318 --          fi:          FailInfo;
17 1319 --
17 1320 --      {If reopen attempt fails, make sure original error gets through.}
17 1321 --      {-----+
17 1322 --      |  Hd1FailFailed  |
17 1323 --      +-----+}
17 1324 -- C  PROCEDURE Hd1FailFailed(newError: OSErr; newMessage: LONGINT);
17 1325 -- 0- C  BEGIN
17 1326 --      Failure(error, message);
17 1327 -- 0 C  END;
17 1328 --
17 1329 -- 0- B  BEGIN
17 1330 --      Hd1SvTemp(error, message);
17 1331 --      IF fSaveExists AND NOT makingCopy THEN
17 1332 --          BEGIN
17 1333 --              CatchFailures(fi, Hd1FailFailed);
17 1334 --              FailOSErr(OpenAFile(fTitle,          fVolRefNum,
17 1335 --                                  fDataOpen,       fRsrcOpen,
17 1336 --                                  fDataPerm,       fRsrcPerm,
17 1337 --                                  dataRefnum,      rsrcRefnum));
17 1338 --              Success(fi);
17 1339 --              fDataRefnum := dataRefnum;
17 1340 --              fRsrcRefnum := rsrcRefnum;
17 1341 --          END;
17 1342 -- 0 B  END;
17 1343 --
17 1344 -- 0- A  BEGIN
17 1345 --      validInfo := GetSaveInfo(itsCmdNumber, copyFInfo, cInfo);
17 1346 --
17 1347 --      GetTempName(tmpName);
17 1348 --
17 1349 --      cInfo.ioNamePtr := @tmpName;
17 1350 --      cInfo.ioVRefnum := volRefnum;
17 1351 --
17 1352 --      MakeNewCopy(makingCopy, validInfo, cInfo);
17 1353 --
17 1354 --      CatchFailures(fi, Hd1SvTemp);
17 1355 --
17 1356 --      {Tell document that the file is going away}
17 1357 --      IF NOT makingCopy THEN
17 1358 --          FreeFile;
17 1359 --
17 1360 --      Success(fi);
17 1361 --
17 1362 --      CatchFailures(fi, Hd1SaveFailed);
17 1363 --
17 1364 --      {Delete the old copy if it exists.}
17 1365 --      err := DeleteFile(@fileName, volRefnum);
17 1366 --      IF err <> noErr THEN
17 1367 --          IF err <> fnfErr THEN
17 1368 --              Failure(err, 0);
17 1369 --
17 1370 --      FailOSErr(Rename(tmpName, volRefnum, fileName));
17 1371 --
17 1372 --      Success(fi);
17 1373 -- 0 A  END;
17 1374 --
17 1375 --
17 1376 --      {$$ Mares}
17 1377 --      {-----+
17 1378 --      |  SetTitle      |
17 1379 --      +-----+}
17 1380 -- A  PROCEDURE TDocument.SetTitle(aTitle: Str255);
17 1381 --
17 1382 --      {-----+
17 1383 --      |  InstallTitle  |
17 1384 --      +-----+}
17 1385 -- B  PROCEDURE InstallTitle(aWindow: TWindow);

```

```

17 1386 0- B      BEGIN
17 1387 --      aWindow.SetTitleForDoc(aTitle);
17 1388 -0 B      END;
17 1389 0- A      BEGIN
17 1390 --      fTitle := aTitle;
17 1391 --      ForAllWindowsDo(InstallTitle);
17 1392 -0 A      END;
17 1393 --
17 1394 --
17 1395 --      {$S MAWriteFile}
17 1396 --      {-----+
17 1397 --      |   SFPutParams
17 1398 --      +-----}
17 1399 -- A      PROCEDURE TDocument.SFPutParams(itsCmdNumber: CmdNumber; VAR dlgID: INTEGER;
17 1400 --      VAR where: Point; VAR defaultName, prompt: Str255; VAR dlgHook, filterProc: ProcPtr);
17 1401 --      CONST kStdSFWhere = 100 * $10000 + 100;
17 1402 --      VAR idx:      INTEGER;
17 1403 --
17 1404 0- A      BEGIN
17 1405 --      dlgID := putDlgID;                                (putDlgID defined by Standard File)
17 1406 --      where := Point(kStdSFWhere);
17 1407 --
17 1408 1-      CASE itsCmdNumber OF
17 1409 --          cSave, cSaveAs: idx := bzSaveAs;
17 1410 --          cSaveCopy:      idx := bzSaveCopy;
17 1411 --          OTHERWISE      idx := 0;
17 1412 -1      END;
17 1413 --
17 1414 --      IF idx = 0 THEN
17 1415 --          prompt := ''
17 1416 --      ELSE
17 1417 --          GetIndString(prompt, kIDBuzzString, idx);
17 1418 --
17 1419 --      dlgHook := NIL;
17 1420 --      filterProc := NIL;
17 1421 -0 A      END;
17 1422 --
17 1423 --
17 1424 --      {$S MAREadFile}
17 1425 --      {-----+
17 1426 --      |   ShowReverted
17 1427 --      +-----}
17 1428 -- A      PROCEDURE TDocument.ShowReverted;
17 1429 --      {-----+
17 1430 --      |   RevertView
17 1431 --      +-----}
17 1432 -- B      PROCEDURE RevertView(aView: TView);
17 1433 0- B      BEGIN
17 1434 --      aView.ShowReverted;
17 1435 -0 B      END;
17 1436 0- A      BEGIN
17 1437 --      ForAllViewsDo(RevertView);
17 1438 -0 A      END;
17 1439 --
17 1440 --
17 1441 --      {$S MAOpen}
17 1442 --      {-----+
17 1443 --      |   ShowWindows
17 1444 --      +-----}
17 1445 -- A      PROCEDURE TDocument.ShowWindows;
17 1446 --      {-----+
17 1447 --      |   ShowAWindow
17 1448 --      +-----}
17 1449 -- B      PROCEDURE ShowAWindow(aWindow: TWindow);
17 1450 0- B      BEGIN
17 1451 --      IF aWindow.fOpenInitially THEN
17 1452 --          aWindow.Open;
17 1453 -0 B      END;
17 1454 0- A      BEGIN
17 1455 --      ForAllWindowsDo(ShowAWindow);
17 1456 -0 A      END;
17 1457 --
17 1458 --
17 1459 --      {$S MAOpen}
17 1460 --      {-----+
17 1461 --      |   UntitledName
17 1462 --      +-----}
17 1463 -- A      PROCEDURE TDocument.UntitledName(VAR noName: Str255);
17 1464 --      VAR preInsert:  INTEGER;
17 1465 --      constChars:  INTEGER;
17 1466 --      num:          Str255;
17 1467 0- A      BEGIN
17 1468 --      GetIndString(noName, kIDBuzzString, bzUntitled);
17 1469 --      IF ParseTitleTemplate(noName, preInsert, constChars) THEN
17 1470 1-      BEGIN
17 1471 --          NumToString(gNumUntitled, num);
17 1472 --
17 1473 --          IF SubstituteInTitle(noName, num, preInsert, constChars) THEN
17 1474 --              gNumUntitled := gNumUntitled + 1;
17 1475 -1      END;
17 1476 -0 A      END;
17 1477 --
17 1478 --
17 1479 --      {$IFC qDebug}
17 1480 --      {$S MAFields}
17 1481 --      {-----+
17 1482 --      |   Fields
17 1483 --      +-----}
17 1484 -- A      PROCEDURE TDocument.Fields (PROCEDURE DoToField (fieldName: Str255;

```



```
17 1485 -- fieldAddr: Ptr;
17 1486 -- fieldType: INTEGER));
17 1487 --
17 1488 0- A BEGIN
17 1489 -- DoToField('TDocument', NIL, bClass);
17 1490 -- DoToField('fWindowList', @fWindowList, bObject);
17 1491 -- DoToField('fViewList', @fViewList, bObject);
17 1492 -- DoToField('fChangeCount', @fChangeCount, bLongInt);
17 1493 -- DoToField('fDocPrintHandler', @fDocPrintHandler, bObject);
17 1494 -- DoToField('fSavePrintInfo', @fSavePrintInfo, bBoolean);
17 1495 -- DoToField('fSharePrintInfo', @fSharePrintInfo, bBoolean);
17 1496 -- DoToField('fPrintInfo', @fPrintInfo, bHandle);
17 1497 -- DoToField('fTitle', @fTitle, bString);
17 1498 -- DoToField('fFileType', @fFileType, bOSType);
17 1499 -- DoToField('fCreator', @fCreator, bOSType);
17 1500 -- DoToField('fVolRefNum', @fVolRefNum, bInteger);
17 1501 -- DoToField('fModDate', @fModDate, bLongInt);
17 1502 -- DoToField('fReopenAlert', @fReopenAlert, bBoolean);
17 1503 -- DoToField('fSaveExists', @fSaveExists, bBoolean);
17 1504 -- DoToField('fCommitOnSave', @fCommitOnSave, bBoolean);
17 1505 -- DoToField('fUsesDataFork', @fUsesDataFork, bBoolean);
17 1506 -- DoToField('fUsesRsrcFork', @fUsesRsrcFork, bBoolean);
17 1507 -- DoToField('fDataOpen', @fDataOpen, bBoolean);
17 1508 -- DoToField('fRsrcOpen', @fRsrcOpen, bBoolean);
17 1509 -- DoToField('fDataPerm', @fDataPerm, bInteger);
17 1510 -- DoToField('fRsrcPerm', @fRsrcPerm, bInteger);
17 1511 -- DoToField('fDataRefNum', @fDataRefNum, bInteger);
17 1512 -- DoToField('fRsrcRefNum', @fRsrcRefNum, bInteger);
17 1513 -- DoToField('fSaveInPlace', @fSaveInPlace, bByte);
17 1514 -- INHERITED Fields(DoToField);
17 1515 -0 A END (TDocument.Fields);
17 1516 -- {$ENDC}
13 3005 -- {$I UMacApp.TView.p}
```

```

18 1 --      {$P}
18 2 --      { UMacApp.TView.p }
18 3 --      { Copyright © 1984-1988 Apple Computer Inc. All rights reserved. }
18 4 --
18 5 --      (*****
18 6 --      (*      G L O B A L      R O U T I N E S      *)
18 7 --      (*****
18 8 --
18 9 --      {$IFC qDebug}
18 10 --      {$S MAdEbug}
18 11 --      {-----+
18 12 --      | WriteFocus                                |
18 13 --      +-----}
18 14 -- A PROCEDURE WriteFocus;
18 15 0- A BEGIN
18 16 --      WrLblVPt( ' gLongOffset', gLongOffset);
18 17 --      WRITELN;
18 18 --      WrLblRect('      portRect', thePort^.portRect);
18 19 --      WRITELN;
18 20 --      WrLblRect('      visRgn', thePort^.visRgn^.rgnbBox);
18 21 --      WRITELN;
18 22 --      WrLblRect('      clipRgn', thePort^.clipRgn^.rgnbBox);
18 23 --      WRITELN;
18 24 0- A END {WriteFocus};
18 25 --      {$ENDC}
18 26 --
18 27 --      {$S MAdEbug}
18 28 --      {-----+
18 29 --      | StripLong                                |
18 30 --      +-----}
18 31 -- A FUNCTION StripLong (address: UNIV Ptr): LONGINT;
18 32 --
18 33 0- A BEGIN
18 34 --      StripLong := LONGINT(StripAddress(address));
18 35 0- A END;
18 36 --
18 37 --      {$S MAdEbug}
18 38 --      {-----+
18 39 --      | NewViewRsrc                                | Create a new view resource
18 40 --      +-----}
18 41 -- A FUNCTION NewViewRsrc (VAR p: UNIV Ptr): ViewRsrcHndl;
18 42 --
18 43 --      VAR
18 44 --          aHandle:      ViewRsrcHndl;
18 45 --
18 46 0- A BEGIN
18 47 --      { ??? Should this be doing a NewPermHandle here ??? }
18 48 --      aHandle := ViewRsrcHndl(NewHandle(kViewRsrcExpandAmt));
18 49 --      MoveHHI(Handle(aHandle));
18 50 --      HLock(Handle(aHandle));
18 51 --      FailMemError;
18 52 --      WITH aHandle^^ DO
18 53 1-          BEGIN
18 54 --              numViews := 0;
18 55 --              p := @theViews;
18 56 1-          END;
18 57 --      NewViewRsrc := aHandle;
18 58 0- A END;
18 59 --
18 60 --      {$S MAdEbug}
18 61 --      {-----+
18 62 --      | DoneViewRsrc                                | Size view resource handle back to real size
18 63 --      +-----}
18 64 -- A PROCEDURE DoneViewRsrc (viewRsrc: UNIV Handle; lastPtr: UNIV LONGINT);
18 65 --
18 66 0- A BEGIN
18 67 --      HUnlock(viewRsrc);
18 68 --      SetHandleSize(viewRsrc, StripLong(lastPtr) - StripLong(viewRsrc));
18 69 0- A END;
18 70 --
18 71 --      {$S MAOpen}
18 72 --      {-----+
18 73 --      | OffsetPtr                                |
18 74 --      +-----}
18 75 -- A PROCEDURE OffsetPtr (VAR p: UNIV LONGINT; offset: LONGINT);
18 76 --
18 77 0- A BEGIN
18 78 --      p := p + offset;
18 79 --      IF ODD(p) THEN
18 80 --          p := p + 1;
18 81 0- A END {OffsetPtr};
18 82 --
18 83 --      {$S MAOpen}
18 84 --      {-----+
18 85 --      | OffsetPtrWStr                                |
18 86 --      +-----}
18 87 -- A PROCEDURE OffsetPtrWStr (VAR p: UNIV LONGINT; offset: LONGINT);
18 88 --
18 89 0- A BEGIN
18 90 --      OffsetPtr(p, offset - 255 + LENGTH(StringPtr(p + offset - 256)^));
18 91 0- A END {OffsetPtrWStr};
18 92 --
18 93 --      {$S MAdEbug}
18 94 --      {-----+
18 95 --      | ExpandPtr                                |
18 96 --      +-----}
18 97 -- A FUNCTION ExpandPtr (viewRsrc: UNIV Handle; VAR p: UNIV LONGINT; offset: LONGINT): Ptr;
18 98 --

```

```

18 99 -- VAR
18 100 --     oldOffset:    LONGINT;
18 101 --     rsrcSize:     Size;
18 102 --     desiredEnd:   LONGINT;
18 103 --     rsrcBase:     LONGINT;
18 104 --     currentPtr:   LONGINT;
18 105 --
18 106 0- A BEGIN
18 107 --     rsrcSize := GetHandleSize(viewRsrc);
18 108 --     rsrcBase := StripLong(viewRsrc^);
18 109 --     currentPtr := StripLong(p);
18 110 --     IF ODD(offset) THEN
18 111 --         offset := offset + 1;
18 112 --     desiredEnd := currentPtr + offset + SIZEOF(INTEGER);
18 113 --
18 114 --     IF desiredEnd >= rsrcBase + rsrcSize THEN
18 115 1- BEGIN
18 116 --         { This appropriation logic might need some re-examination.  If the size of the added
18 117 --           template is larger than the minimum amount, then simply the size is added.  If
18 118 --           the handle is already near to being full, this won't help for the next allocation.
18 119 --           Maybe it should use a hysteresis?_ }
18 120 --         oldOffset := currentPtr - rsrcBase;
18 121 --         HUnlock(viewRsrc);
18 122 --         SetHandleSize(viewRsrc, rsrcSize + MAX(kViewRsrcExpandAmt, offset));
18 123 --         FailMemError;
18 124 --         MoveHHI(viewRsrc);
18 125 --         HLock(viewRsrc);
18 126 --         p := LONGINT(viewRsrc^) + oldOffset;
18 127 -1 END;
18 128 --     ExpandPtr := Ptr(p);
18 129 --     OffsetPtr(p, offset);
18 130 -0 A END;
18 131 --
18 132 --     {$S MAWriteRes}
18 133 --     {-----+
18 134 --     | ExpandPtrWStr
18 135 --     +-----+
18 136 -- A FUNCTION ExpandPtrWStr (viewRsrc: UNIV Handle; VAR p: UNIV LONGINT;
18 137 --     offset, len: LONGINT): Ptr;
18 138 0- A BEGIN
18 139 --     ExpandPtrWStr := ExpandPtr(viewRsrc, p, offset - 255 + len);
18 140 -0 A END;
18 141 --
18 142 --     {$S MAMain}
18 143 --     {-----+
18 144 --     | PtIsVisible
18 145 --     +-----+
18 146 -- A FUNCTION PtIsVisible (pt: Point): BOOLEAN;
18 147 0- A BEGIN
18 148 --     PtIsVisible := PtInRgn(pt, thePort^.visRgn) & PtInRgn(pt, thePort^.clipRgn);
18 149 -0 A END (PtIsVisible);
18 150 --
18 151 --     {$S MAREs}
18 152 --     {-----+
18 153 --     | VisibleRect
18 154 --     +-----+
18 155 -- A PROCEDURE VisibleRect (VAR r: Rect);
18 156 --
18 157 0- A BEGIN
18 158 --     IF NOT gDrawingPictScrap THEN
18 159 1- BEGIN
18 160 --     {$IFC qDebug}
18 161 --         UseTempRgn('VisibleRect');
18 162 --     {$ENDC}
18 163 --         RectRgn(gTempRgn, r);
18 164 --         SectRgn(gTempRgn, thePort^.clipRgn, gTempRgn);
18 165 --         r := gTempRgn^^.rgnBBox;
18 166 --     {$IFC qDebug}
18 167 --         DoneWithTempRgn;
18 168 --     {$ENDC}
18 169 -1 END;
18 170 -0 A END (VisibleRect);
18 171 --
18 172 --     (*****
18 173 --     (*      T v i e w
18 174 --     (*****
18 175 --     {-----+
18 176 --     | IView
18 177 --     +-----+
18 178 --     {$S MAOpen}
18 179 - A PROCEDURE TView.IView (itsDocument: TDocument; itsSuperview: TView; itsLocation: VPoint; itsSize: VPoint;
18 180 --     itsHSizeDet, itsVSizeDet: SizeDeterminer);
18 181 --
18 182 --     VAR
18 183 --         fi:      FailInfo;
18 184 --
18 185 -- B PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
18 186 0- B BEGIN
18 187 --     Free;
18 188 -0 B END;
18 189 --
18 190 0- A BEGIN
18 191 --     fSuperview := itsSuperview;
18 192 --     fSubviews := NIL;
18 193 --     fDocument := itsDocument;
18 194 --     fLocation := itsLocation;
18 195 --     fSize := itsSize;
18 196 --     fSizeDeterminer[h] := itsHSizeDet;
18 197 --     fSizeDeterminer[v] := itsVSizeDet;

```

```

18 198 --      fHLDdesired := hloff;
18 199 --      fIdentifier := kNoIdentifier;
18 200 --      fShown := TRUE;
18 201 --      fViewEnabled := TRUE;
18 202 --      fPrintHandler := NIL;
18 203 --
18 204 --      IEvtHandler(itsSuperview);
18 205 --
18 206 --      CatchFailures(fi, HandleFailure);
18 207 --      IF itsSuperview <> NIL THEN
18 208 --          itsSuperview.AddSubview(SELF);
18 209 --      IF itsdocument <> NIL THEN
18 210 --          itsdocument.AddView(SELF);
18 211 --      Success(fi);
18 212 -- A  END (TView.IView);
18 213 --
18 214 --
18 215 --      {-----+
18 216 --      | IRes                                     |
18 217 --      +-----+
18 218 --      {$$ MAOpen}
18 219 -- A  PROCEDURE TView.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr);
18 220 --
18 221 -- A  BEGIN
18 222 --      WITH ViewTemplatePtr(itsParams)^ DO
18 223 --      BEGIN
18 224 --          IView(itsDocument, itsSuperview, itsLocation, itsSize,
18 225 --              SizeDeterminer(itsHSizeDet), SizeDeterminer(itsVSizeDet));
18 226 --          fShown := isShown;
18 227 --          fViewEnabled := isEnabled;
18 228 --          fHLDdesired := hloff;
18 229 --          fIdentifier := thisViewID;
18 230 --          OffsetPtrWStr(itsParams, SIZEOF(ViewTemplate));
18 231 --      END;
18 232 -- A  END;
18 233 --
18 234 --
18 235 --      {-----+
18 236 --      | WRes                                     |
18 237 --      +-----+
18 238 --      {$$ MAWriteRes}
18 239 -- A  PROCEDURE TView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr);
18 240 --      VAR
18 241 --          vwPtr:          ViewTemplatePtr;
18 242 --
18 243 -- A  BEGIN
18 244 --      vwPtr := ViewTemplatePtr(ExpandPtrWStr(theResource, itsParams,
18 245 --          SIZEOF(ViewTemplate), LENGTH(gWResType)));
18 246 --
18 247 --      WITH vwPtr^ DO
18 248 --      BEGIN
18 249 --          IF fSuperview <> NIL THEN
18 250 --              itsParentID := fSuperview.fIdentifier
18 251 --          ELSE
18 252 --              itsParentID := kNoIdentifier;
18 253 --          thisViewID := fIdentifier;
18 254 --          itsLocation := fLocation;
18 255 --          itsSize := fSize;
18 256 --          itsVSizeDet := Integer8(fSizeDeterminer[v]);
18 257 --          itsHSizeDet := Integer8(fSizeDeterminer[h]);
18 258 --          isShown := fShown;
18 259 --          isEnabled := fViewEnabled;
18 260 --          itsSignature := gWResSignature;
18 261 --          (* itsType := gWResType; *)
18 262 --          CopyStr255(gWResType, PRStr(itsType));
18 263 --      END;
18 264 --
18 265 --      { Everybody gets this far - bump number of views in resource by 1 }
18 266 --      WITH theResource^^ DO
18 267 --          numViews := numViews + 1;
18 268 -- A  END;
18 269 --
18 270 --
18 271 --      {-----+
18 272 --      | WriteRes                                 |
18 273 --      +-----+
18 274 --      {$$ MAWriteRes}
18 275 -- A  PROCEDURE TView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr);
18 276 -- A  BEGIN
18 277 --      gWResSignature := 'view'; gWResType := 'TView';
18 278 --      WRes(theResource, itsParams);
18 279 -- A  END;
18 280 --
18 281 --
18 282 --      {-----+
18 283 --      | Free                                     |
18 284 --      +-----+
18 285 --      {$$ MAClose}
18 286 -- A  PROCEDURE TView.Free;
18 287 --
18 288 -- A  BEGIN
18 289 --      IF fSubViews <> NIL THEN
18 290 --          fSubViews.FreeList;
18 291 --          fSubViews := NIL;
18 292 --      IF fSuperview <> NIL THEN
18 293 --          fSuperview.RemoveSubview(SELF);
18 294 --      IF fdocument <> NIL THEN
18 295 --          fdocument.DeleteView(SELF);
18 296 --      IF fPrintHandler <> NIL THEN

```

```

18 297 --      fPrintHandler.Free;
18 298 --
18 299 --      INHERITED Free;
18 300 -- A  END (TView.Free);
18 301 --
18 302 --      {-----+
18 303 --      |  Activate      |
18 304 --      +-----+
18 305 --      {$S MAActivate}
18 306 -- A  PROCEDURE TView.Activate (entering: BOOLEAN);
18 307 --
18 308 --      {-----+
18 309 --      |  ActivateSubView  |
18 310 --      +-----+
18 311 -- B  PROCEDURE ActivateSubView (theSubView: TView);
18 312 -- B  BEGIN
18 313 --      theSubView.Activate(entering);
18 314 -- B  END;
18 315 --
18 316 -- A  BEGIN
18 317 --      IF Focus THEN ;
18 318 --      IF entering THEN
18 319 --          BEGIN
18 320 --              DoHighlightSelection(fHLDdesired, h1On);
18 321 --              fHLDdesired := h1On;
18 322 --          END
18 323 --      ELSE
18 324 --          BEGIN
18 325 --              DoHighlightSelection(fHLDdesired, h1Dim);
18 326 --              fHLDdesired := h1Dim;
18 327 --          END;
18 328 --      EachSubView(ActivateSubView);
18 329 -- A  END (TView.Activate);
18 330 --
18 331 --      {-----+
18 332 --      |  AddSubView      |
18 333 --      +-----+
18 334 --      {$S MAOpen}
18 335 -- A  PROCEDURE TView.AddSubView (theSubView: TView);
18 336 --      VAR
18 337 --          itsGrafPort:      GrafPtr;
18 338 --
18 339 -- A  BEGIN
18 340 --      IF theSubView <> NIL THEN
18 341 --          BEGIN
18 342 --              IF fSubViews = NIL THEN
18 343 --                  BEGIN
18 344 --                      fSubViews := NewList;
18 345 --                      ($IFC qDebug)
18 346 --                          fSubViews.SetEltType('TView');
18 347 --                      ($ENDC)
18 348 --                  END;
18 349 --              ($IFC FALSE)
18 350 --                  IF fSubViews.GetItemNumber(theSubView) <> 0 THEN
18 351 --                      ProgramBreak('Attempting to duplicate a subview in AddSubView.');
```

```

18 396 --      CntlFrame =      [adnLineTop, adnLineLeft, adnLineBottom, adnLineRight];
18 397 --      VAR
18 398 --      shadowed:      BOOLEAN;
18 399 --      savedPenState:  PenState;
18 400 --
18 401 --      {-----+
18 402 --      | XFrameOval      |
18 403 --      +-----}
18 404 -- B      PROCEDURE XFrameOval;
18 405 0- B      BEGIN
18 406 --          FrameOval(area);
18 407 -0 B      END;
18 408 --
18 409 --      {-----+
18 410 --      | XFrameRRect     |
18 411 --      +-----}
18 412 -- B      PROCEDURE XFrameRRect;
18 413 0- B      BEGIN
18 414 --          FrameRoundRect(area, 16, 16);
18 415 -0 B      END;
18 416 --
18 417 --      {-----+
18 418 --      | DrawWithShadow  |
18 419 --      +-----}
18 420 -- B      PROCEDURE DrawWithShadow (PROCEDURE DrawCmd);
18 421 --      LABEL
18 422 --      1;
18 423 --      VAR
18 424 --          drawRgn:      RgnHandle;
18 425 --          shadowRgn:     RgnHandle;
18 426 --          fi:           FailInfo;
18 427 --
18 428 --      {-----+
18 429 --      | HandleFailure   |
18 430 --      +-----}
18 431 -- C      PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
18 432 0- C      BEGIN
18 433 --          { If we fail allocating memory for the regions, dispose of the regions if
18 434 --            they exist and continue}
18 435 --          GOTO 1;
18 436 -0 C      END;
18 437 --
18 438 0- B      BEGIN
18 439 --          IF shadowed THEN
18 440 1-          BEGIN
18 441 --              shadowRgn := NIL;          {In case we creating drawRgn fails}
18 442 --              CatchFailures(fi, HandleFailure);
18 443 --              drawRgn := MakeNewRgn;
18 444 --              shadowRgn := MakeNewRgn;
18 445 --
18 446 --              OpenRgn;
18 447 --              DrawCmd;
18 448 --              CloseRgn(drawRgn);
18 449 --
18 450 --              CopyRgn(drawRgn, shadowRgn);
18 451 --              OffsetRgn(shadowRgn, itsPenSize.h, itsPenSize.v);
18 452 --              DiffRgn(shadowRgn, drawRgn, shadowRgn);
18 453 --              PaintRgn(shadowRgn);
18 454 --              Success(fi);
18 455 --          1:
18 456 --              IF drawRgn <> NIL THEN
18 457 --                  DisposeRgn(drawRgn);
18 458 --              IF shadowRgn <> NIL THEN
18 459 --                  DisposeRgn(shadowRgn);
18 460 -1          END;
18 461 --          DrawCmd;
18 462 -0 B      END;
18 463 --
18 464 --      {-----+
18 465 --      | DrawLine        |
18 466 --      +-----}
18 467 -- B      PROCEDURE DrawLine(vhs: VHSelect; lh, lv, lto: INTEGER);
18 468 0- B      BEGIN
18 469 --          MoveTo(lh, lv);
18 470 1-          CASE vhs OF
18 471 --              h: Line(lto-lh, 0);
18 472 --              v: Line(0, lto-lv);
18 473 -1          END;
18 474 -0 B      END;
18 475 --
18 476 0- A      BEGIN
18 477 --          IF itsAdornment <> [] THEN          { Only do this if adorning is asked for }
18 478 1-          BEGIN
18 479 --              GetPenState(savedPenState);
18 480 --              PenNormal;
18 481 --              PenSize(itsPenSize.h, itsPenSize.v);
18 482 --
18 483 --              shadowed := adnShadow IN itsAdornment;
18 484 --              IF shadowed THEN
18 485 --                  SubPt(itsPenSize, area.botRight);
18 486 --
18 487 --              IF adnOval IN itsAdornment THEN
18 488 --                  DrawWithShadow(XFrameOval);
18 489 --
18 490 --              IF adnRRect IN itsAdornment THEN
18 491 --                  DrawWithShadow(XFrameRRect);
18 492 --
18 493 --              IF itsAdornment >= CntlFrame THEN { Special case for whole rect! }
18 494 2-              BEGIN

```

```

18 495 --      FrameRect(area);
18 496 --      IF shadowed THEN
18 497 --          WITH area DO
18 498 3-              BEGIN
18 499 --                  DrawLine(h, left + itsPenSize.h, bottom, right);
18 500 --                  DrawLine(v, right, top + itsPenSize.v, bottom);
18 501 -3              END;
18 502 -2          END
18 503 --      ELSE
18 504 --          WITH area DO
18 505 2-              BEGIN
18 506 --                  { No shadows for single lines }
18 507 --                  IF adnLineTop IN itsAdornment THEN
18 508 --                      DrawLine(h, left, top, right);
18 509 --                  IF adnLineLeft IN itsAdornment THEN
18 510 --                      DrawLine(v, left, top, bottom);
18 511 --                  IF adnLineBottom IN itsAdornment THEN
18 512 --                      DrawLine(h, left, bottom - 1, right);
18 513 --                  IF adnLineRight IN itsAdornment THEN
18 514 --                      DrawLine(v, right - 1, top, bottom);
18 515 --                  END;
18 516 --                  SetPenState(savedPenState);
18 517 -0 A      END (TView.Adorn);
18 518 --
18 519 --      {-----+
18 520 --      | AttachPrintHandler |
18 521 --      +-----}
18 522 --      {$S MRes}
18 523 -- A      PROCEDURE TView.AttachPrintHandler (itsPrintHandler: TPrintHandler);
18 524 0- A      BEGIN
18 525 --          fPrintHandler := itsPrintHandler;
18 526 --          DoCheckPrinter;
18 527 -0 A      END (TView.AttachPrintHandler);
18 528 --
18 529 --      {-----+
18 530 --      | BeInPort |
18 531 --      +-----}
18 532 --      {$S MNonRes}
18 533 -- A      PROCEDURE TView.BeInPort (itsPort: GrafPtr);
18 534 --
18 535 --      {-----+
18 536 --      | NotifySubView |
18 537 --      +-----}
18 538 -- B      PROCEDURE NotifySubView (theSubView: TView);
18 539 0- B      BEGIN
18 540 --          theSubView.BeInPort(itsPort);
18 541 -0 B      END;
18 542 --
18 543 0- A      BEGIN
18 544 --          IF fPrintHandler <> NIL THEN
18 545 --              DoCheckPrinter;
18 546 --              EachSubView(NotifySubView);
18 547 -0 A      END (TView.BeInPort);
18 548 --
18 549 --      {-----+
18 550 --      | BeInScroller |
18 551 --      +-----}
18 552 --      {$S MNonRes}
18 553 -- A      PROCEDURE TView.BeInScroller (itsScroller: TScroller);
18 554 --      VAR
18 555 --          scrollLimits:      VPoint;
18 556 --
18 557 0- A      BEGIN
18 558 --          IF itsScroller <> NIL THEN
18 559 1-              BEGIN
18 560 --                  scrollLimits := fSize;
18 561 --                  AddVpt(fLocation, scrollLimits);
18 562 --                  itsScroller.SetScrollLimits(scrollLimits, TRUE);
18 563 -1              END;
18 564 -0 A      END (TView.BeInScroller);
18 565 --
18 566 --      {-----+
18 567 --      | CalcMinSize |
18 568 --      +-----}
18 569 --      {$S MRes}
18 570 -- A      PROCEDURE TView.CalcMinSize (VAR minSize: VPoint);
18 571 0- A      BEGIN
18 572 --          minSize := fSize;
18 573 -0 A      END (TView.CalcMinSize);
18 574 --
18 575 --      {-----+
18 576 --      | ClipFurtherTo |
18 577 --      +-----}
18 578 --      {$S MRes}
18 579 -- A      PROCEDURE TView.ClipFurtherTo(r: Rect; hDeltaOrg, vDeltaOrg: INTEGER);
18 580 --          { Set clipping to: (current clipping) INTERSECT r) OFFSET-BY (hDeltaOrg, vDeltaOrg) }
18 581 --
18 582 0- A      BEGIN
18 583 --          {$IFC qDebug}
18 584 --          UseTempRgn('ClipFurtherTo');
18 585 --          {$ENDC}
18 586 --
18 587 --          RectRgn(gTempRgn, r);
18 588 --          SectRgn(thePort^.clipRgn, gTempRgn, gTempRgn);
18 589 --          OffsetRgn(gTempRgn, hDeltaOrg, vDeltaOrg);
18 590 --          SetClip(gTempRgn);
18 591 --
18 592 --          {$IFC qDebug}
18 593 --          DoneWithTempRgn;

```

```

18 594 --      {$ENDC}
18 595 -0 A  END {TView.ClipFurtherTo};
18 596 --
18 597 --      {-----+
18 598 --      |   Close                               |
18 599 --      +-----}
18 600 --      {$S MAClose}
18 601 -0 A  PROCEDURE TView.Close;
18 602 --
18 603 --      {-----+
18 604 --      |   CloseSubView                       |
18 605 --      +-----}
18 606 -- B    PROCEDURE CloseSubView (theSubView: TView);
18 607 0- B    BEGIN
18 608 --      theSubView.Close;
18 609 -0 B    END;
18 610 --
18 611 0- A  BEGIN
18 612 --      EachSubView(CloseSubView);
18 613 -0 A  END {TView.Close};
18 614 --
18 615 --      {-----+
18 616 --      |   ComputeSize                         |
18 617 --      +-----}
18 618 --      {$S MRes}
18 619 -0 A  PROCEDURE TView.ComputeSize (VAR newSize: VPoint);
18 620 --      VAR
18 621 --          dimension:    VCoordinate;
18 622 --          vhs:          VHSelect;
18 623 --          minSize:      VPoint;
18 624 --
18 625 0- A  BEGIN
18 626 --      CalcMinSize(minSize);
18 627 --
18 628 --      { Adjust view size based on size determiners. }
18 629 --
18 630 --      FOR vhs := v TO h DO
18 631 1-      BEGIN
18 632 2-      CASE fSizeDeterminer[vhs] OF
18 633 --          sizeVariable:
18 634 --              dimension := minSize.vh[vhs];
18 635 --          sizeFixed:
18 636 --              dimension := fSize.vh[vhs];
18 637 --          sizeRelSuperView:
18 638 --              dimension := newSize.vh[vhs];  { newSize set in SuperViewChangedSize }
18 639 --          sizeSuperView:
18 640 --              IF fSuperView <> NIL THEN
18 641 --                  dimension := fSuperView.fSize.vh[vhs];
18 642 --          sizePage:
18 643 --              IF fPrintHandler <> NIL THEN
18 644 --                  dimension := fPrintHandler.fViewPerPage.vh[vhs]
18 645 --      ($IFC qDebug)
18 646 --          ELSE
18 647 --              ProgramBreak('Size determiner of sizePage, but no print handler.')
18 648 --      ($ENDC)
18 649 --      ;
18 650 --      sizeFillPages:
18 651 --          IF fPrintHandler <> NIL THEN
18 652 --              dimension := Min(
18 653 --                  RoundUp(minSize.vh[vhs], fPrintHandler.fViewPerPage.vh[vhs]),
18 654 --                  kMaxCoord)
18 655 --      ($IFC qDebug)
18 656 --          ELSE
18 657 --              ProgramBreak('Size determiner of sizePage, but no print handler.')
18 658 --      ($ENDC)
18 659 --      ;
18 660 -2      END;
18 661 --      newSize.vh[vhs] := dimension;
18 662 -1      END;
18 663 -0 A  END {TView.ComputeSize};
18 664 --
18 665 --      {-----+
18 666 --      |   ContainsClipType                     |
18 667 --      +-----}
18 668 --      {$S MRes}
18 669 -0 A  FUNCTION TView.ContainsClipType (aType: ResType): BOOLEAN;
18 670 --      VAR
18 671 --          offset: LONGINT;
18 672 --
18 673 0- A  BEGIN
18 674 --      ContainsClipType := (GetScrap(NIL, aType, offset) > 0)
18 675 -0 A  END {TView.ContainsClipType};
18 676 --
18 677 --      {-----+
18 678 --      |   ContainsMouse                         |
18 679 --      +-----}
18 680 --      {$S MRes}
18 681 -0 A  FUNCTION TView.ContainsMouse (theMouse: VPoint): BOOLEAN;
18 682 --
18 683 --      VAR
18 684 --          extent:    VRect;
18 685 --
18 686 0- A  BEGIN
18 687 --      GetExtent(extent);
18 688 --      ContainsMouse := IsShown AND PtInVRect(theMouse, extent);
18 689 -0 A  END {TView.ContainsMouse};
18 690 --
18 691 --      {-----+
18 692 --      |   CountSubViews                       |

```



```

18 693 -- +-----+
18 694 -- { $$ MRes }
18 695 -- A FUNCTION TView.CountSubViews: INTEGER;
18 696 0- A BEGIN
18 697 -- IF fSubViews <> NIL THEN
18 698 --     CountSubViews := fSubViews.fSize
18 699 -- ELSE
18 700 --     CountSubViews := 0;
18 701 -0 A END {TView.CountSubViews};
18 702 --
18 703 -- +-----+
18 704 -- | DoBreakFollowing |
18 705 -- +-----+
18 706 -- { $$ MRes }
18 707 -- A FUNCTION TView.DoBreakFollowing (vhs: VHSelect; prevBreak: VCoordinate;
18 708 -- VAR automatic: BOOLEAN): VCoordinate;
18 709 0- A BEGIN
18 710 --     DoBreakFollowing := fPrintHandler.BreakFollowing(vhs, prevBreak, automatic);
18 711 -0 A END {TView.DoBreakFollowing};
18 712 --
18 713 -- +-----+
18 714 -- | DoCalcPageStrips |
18 715 -- +-----+
18 716 -- { $$ MAnonRes }
18 717 -- A PROCEDURE TView.DoCalcPageStrips (VAR pageStrips: Point);
18 718 0- A BEGIN
18 719 --     fPrintHandler.CalcPageStrips(pageStrips);
18 720 -0 A END {TView.DoCalcPageStrips};
18 721 --
18 722 -- { $$ MAnonRes }
18 723 -- +-----+
18 724 -- | DoCalcViewPerPage |
18 725 -- +-----+
18 726 -- { $$ MAnonRes }
18 727 -- A PROCEDURE TView.DoCalcViewPerPage (VAR viewPerPage: VPoint);
18 728 0- A BEGIN
18 729 --     viewPerPage := fSize; (in case fPrintHandler is vacuous)
18 730 --     fPrintHandler.CalcViewPerPage(viewPerPage);
18 731 -0 A END {TView.DoCalcViewPerPage};
18 732 --
18 733 -- +-----+
18 734 -- | DoCheckPrinter |
18 735 -- +-----+
18 736 -- { $$ MRes }
18 737 -- A PROCEDURE TView.DoCheckPrinter;
18 738 0- A BEGIN
18 739 --     fPrintHandler.CheckPrinter
18 740 -0 A END {TView.DoCheckPrinter};
18 741 --
18 742 -- +-----+
18 743 -- | DoChoice |
18 744 -- +-----+
18 745 -- { $$ MRes }
18 746 -- A PROCEDURE TView.DoChoice (origView: TView; itsChoice: INTEGER);
18 747 0- A BEGIN
18 748 --     IF fSuperView <> NIL THEN
18 749 --         fSuperView.DoChoice (origView, itsChoice);
18 750 -0 A END;
18 751 --
18 752 -- +-----+
18 753 -- | DoDrawPrintFeedback |
18 754 -- +-----+
18 755 -- { $$ MRes }
18 756 -- A PROCEDURE TView.DoDrawPrintFeedback (area: Rect);
18 757 0- A BEGIN
18 758 --     IF fPrintHandler <> NIL THEN
18 759 --         fPrintHandler.DrawPrintFeedback(area);
18 760 -0 A END {TView.DoDrawPrintFeedback};
18 761 --
18 762 -- +-----+
18 763 -- | DoDrawPageBreak |
18 764 -- +-----+
18 765 -- { $$ MRes }
18 766 -- A PROCEDURE TView.DoDrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
18 767 -- loc: VCoordinate; automatic: BOOLEAN);
18 768 0- A BEGIN
18 769 --     fPrintHandler.DrawPageBreak(vhs, whichBreak, loc, automatic);
18 770 -0 A END {TView.DoDrawPageBreak};
18 771 --
18 772 -- +-----+
18 773 -- | DoHighlightSelection |
18 774 -- +-----+
18 775 -- { $$ MRes }
18 776 -- A PROCEDURE TView.DoHighlightSelection (fromHL, toHL: HLState);
18 777 0- A BEGIN
18 778 -0 A END {TView.DoHighlightSelection};
18 779 --
18 780 -- +-----+
18 781 -- | DoMenuCommand |
18 782 -- +-----+
18 783 -- { $$ MAselCommand }
18 784 -- A FUNCTION TView.DoMenuCommand (aCmdNumber: CmdNumber): TCommand;
18 785 0- A BEGIN
18 786 1- CASE aCmdNumber OF
18 787 --     cPrViewBase..cPrViewMax,
18 788 --     cPrFileBase..cPrFileMax:
18 789 --         IF fPrintHandler <> NIL THEN
18 790 --             DoMenuCommand := fPrintHandler.DoMenuCommand(aCmdNumber)
18 791 --         ELSE

```

```

18 792 --          DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
18 793 --          OTHERWISE
18 794 --          DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
18 795 --          END;
18 796 -0 A      END (TView.DoMenuCommand);
18 797 --
18 798 --          {-----+
18 799 --          | DoMouseCommand |
18 800 --          +-----}
18 801 --          {$S MASElCommand}
18 802 -- A      FUNCTION TView.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
18 803 --          VAR hysteresis: Point): TCommand;
18 804 0- A      BEGIN
18 805 --          DoMouseCommand := gNoChanges;
18 806 -0 A      END (TView.DoMouseCommand);
18 807 --
18 808 --          {-----+
18 809 --          | DoPagination |
18 810 --          +-----}
18 811 --          {$S MANonRes}
18 812 -- A      PROCEDURE TView.DoPagination;
18 813 0- A      BEGIN
18 814 --          IF fPrintHandler <> NIL THEN
18 815 --              fPrintHandler.RedoPageBreaks
18 816 -0 A      END (TView.DoPagination);
18 817 --
18 818 --          {-----+
18 819 --          | DoPrinterChanged |
18 820 --          +-----}
18 821 --          {$S MANonRes}
18 822 -- A      PROCEDURE TView.DoPrinterChanged;
18 823 0- A      BEGIN
18 824 --          IF fPrintHandler <> NIL THEN
18 825 --              fPrintHandler.PrinterChanged;
18 826 -0 A      END (TView.DoPrinterChanged);
18 827 --
18 828 --          {-----+
18 829 --          | DoSetPageOffset |
18 830 --          +-----}
18 831 --          {$S MAPrint}
18 832 -- A      PROCEDURE TView.DoSetPageOffset (coord: VPoint);
18 833 0- A      BEGIN
18 834 --          fPrintHandler.SetPageOffset(coord);
18 835 -0 A      END (TView.DoSetPageOffset);
18 836 --
18 837 --          {-----+
18 838 --          | DoSetCursor |
18 839 --          +-----}
18 840 --          {$S MAREs}
18 841 -- A      FUNCTION TView.DoSetCursor (localPoint: Point;
18 842 --          cursorRgn: RgnHandle): BOOLEAN;
18 843 0- A      BEGIN
18 844 --          DoSetCursor := FALSE;
18 845 -0 A      END (TView.DoSetCursor);
18 846 --
18 847 --          {-----+
18 848 --          | DoSetupMenus |
18 849 --          +-----}
18 850 --          {$S MAREs}
18 851 -- A      PROCEDURE TView.DoSetupMenus;
18 852 0- A      BEGIN
18 853 --          INHERITED DoSetupMenus;
18 854 --
18 855 --          IF fPrintHandler <> NIL THEN
18 856 --              fPrintHandler.DoSetupMenus;
18 857 -0 A      END;
18 858 --
18 859 --          {-----+
18 860 --          | Draw |
18 861 --          +-----}
18 862 --          {$S MAREs}
18 863 -- A      PROCEDURE TView.Draw (area: Rect);
18 864 0- A      BEGIN
18 865 -0 A      END (TView.Draw);
18 866 --
18 867 --          {-----+
18 868 --          | DrawContents |
18 869 --          +-----}
18 870 --          {$S MAREs}
18 871 -- A      PROCEDURE TView.DrawContents;
18 872 --          VAR
18 873 --              longOffset: VPoint;
18 874 --              visRect: Rect;
18 875 --              savedClip: RgnHandle;
18 876 --              savedOrigin: Point;
18 877 --              savedLongOffset: VPoint;
18 878 --              displaying: BOOLEAN;
18 879 --
18 880 --          {-----+
18 881 --          | DrawSubView |
18 882 --          +-----}
18 883 -- B      PROCEDURE DrawSubView (theSubView: TView);
18 884 --          VAR
18 885 --              itsFrame: VRect;
18 886 --              itsQDFrame: Rect;
18 887 0- B      BEGIN
18 888 --          IF theSubView.IsShown THEN
18 889 1-          BEGIN
18 890 --              SetOrigin(savedOrigin.h, savedOrigin.v); { Restore superview's focus. }

```

```

18 891 --      SetClip(savedClip);
18 892 --      gLongOffset := savedLongOffset;
18 893 --      gFocusedView := SELF;
18 894 --
18 895 --      theSubView.GetFrame(itsFrame);                ( See if this subview is visible. )
18 896 --      OffsetVRect(itsFrame, -longOffset.h, -longOffset.v);
18 897 --      VRectToRect(itsFrame, itsQDFrame);
18 898 --      IF (NOT displaying) | RectInRgn(itsQDFrame, thePort^.visRgn) THEN
18 899 --          theSubView.DrawContents;
18 900 --      END;
18 901 --0 B      END;
18 902 --
18 903 --0 A      BEGIN
18 904 --          IF Focus THEN
18 905 --              BEGIN
18 906 --                  GetVisibleRect(visRect);
18 907 --                  displaying := NOT (gPrinting OR gDrawingPictScrap);
18 908 --                  IF (NOT displaying) | SectRect(visRect, thePort^.visRgn^.rgnbBox, visRect) THEN
18 909 --                      BEGIN
18 910 --                          Draw(visRect);
18 911 --                      END;
18 912 --                  IF displaying THEN
18 913 --                      BEGIN
18 914 --                          DoHighlightSelection(hlOff, fhLDesired);
18 915 --                          IF fPrintHandler <> NIL THEN
18 916 --                              DoDrawPrintFeedback(visRect);
18 917 --                          END;
18 918 --                      END;
18 919 --                  IF CountSubViews > 0 THEN
18 920 --                      BEGIN
18 921 --                          longOffset := gLongOffset;
18 922 --                          savedClip := MakeNewRgn;                ( Save the superview's focus. )
18 923 --                          savedOrigin := thePort^.portRect.topLeft;
18 924 --                          savedLongOffset := gLongOffset;
18 925 --                          GetClip(savedClip);
18 926 --                          EachSubView(DrawSubView);
18 927 --                          DisposeRgn(savedClip);
18 928 --                      END;
18 929 --                  END;
18 930 --              END;
18 931 --0 A      END (TView.DrawContents);
18 932 --
18 933 --      {-----+
18 934 --      | EachSubView |
18 935 --      +-----+
18 936 --      {$$ MRes}
18 937 --0 A      PROCEDURE TView.EachSubView (PROCEDURE DoToSubView (theSubView: TView));
18 938 --0 A      BEGIN
18 939 --          IF fSubViews <> NIL THEN
18 940 --              fSubViews.Each(DoToSubView);
18 941 --0 A      END (TView.EachSubView);
18 942 --
18 943 --      {-----+
18 944 --      | FindSubView |
18 945 --      +-----+
18 946 --      {$$ MRes}
18 947 --0 A      FUNCTION TView.FindSubView (itsIdentifier: IDType): TView;
18 948 --      VAR
18 949 --          foundView:    TView;
18 950 --          dummyView:    TView;
18 951 --
18 952 --      {-----+
18 953 --      | TestSubID |
18 954 --      +-----+
18 955 --0 B      FUNCTION TestSubID (theSubView: TView): BOOLEAN;
18 956 --
18 957 --0 B      BEGIN
18 958 --          IF LONGINT(theSubView.fIdentifier) = LONGINT(itsIdentifier) THEN
18 959 --              foundView := theSubView
18 960 --          ELSE
18 961 --              dummyView := theSubView.FirstSubViewThat(TestSubID);
18 962 --              TestSubID := foundView <> NIL;
18 963 --0 B      END;
18 964 --
18 965 --0 A      BEGIN
18 966 --          IF LONGINT(fIdentifier) = LONGINT(itsIdentifier) THEN
18 967 --              foundView := SELF
18 968 --          ELSE
18 969 --              BEGIN
18 970 --                  foundView := NIL;
18 971 --                  dummyView := FirstSubViewThat(TestSubID);
18 972 --              END;
18 973 --          {$IFC qDebug}
18 974 --          IF foundView = NIL THEN
18 975 --              BEGIN
18 976 --                  WRITELN('Unable to find subview: ', itsIdentifier, '');
18 977 --                  IF gIntenseDebugging THEN
18 978 --                      ProgramBreak('!');
18 979 --              END;
18 980 --          {$ENDC}
18 981 --          FindSubView := foundView;
18 982 --0 A      END;
18 983 --
18 984 --      {-----+
18 985 --      | FirstSubViewThat |
18 986 --      +-----+
18 987 --      {$$ MRes}
18 988 --0 A      FUNCTION TView.FirstSubViewThat (FUNCTION TestSubView (theSubView: TView): BOOLEAN): TView;
18 989 --0 A      BEGIN

```

```

18 990 --      IF fSubViews <> NIL THEN
18 991 --          FirstSubviewThat := TView(fSubViews.FirstThat(TestSubview))
18 992 --      ELSE
18 993 --          FirstSubviewThat := NIL;
18 994 --0 A  END (TView.FirstSubviewThat);
18 995 --
18 996 --      {-----+
18 997 --      | Focus
18 998 --      +-----}
18 999 --      {$S Mares}
18 1000 -- A  FUNCTION TView.Focus: BOOLEAN;
18 1001 --      VAR
18 1002 --          relOrigin:    VPoint;
18 1003 --          qdFrame:      Rect;
18 1004 --          aRect:         Rect;
18 1005 --          viewRect:      VRect;
18 1006 --          visRect:       VRect;
18 1007 --          myFrame:       VRect;
18 1008 --          vhs:           VHSelect;
18 1009 --
18 1010 --0 A  BEGIN
18 1011 --      Focus := TRUE;
18 1012 --
18 1013 --      IF gFocusedView = SELF THEN
18 1014 --          EXIT(Focus);
18 1015 --
18 1016 --      IF (gCurrPrintHandler <> NIL) & (gCurrPrintHandler.fView = SELF) THEN
18 1017 --1-      BEGIN
18 1018 --          gCurrPrintHandler.FocusOnInterior;
18 1019 --          gFocusedView := SELF;
18 1020 --          EXIT(Focus);
18 1021 --1-      END;
18 1022 --
18 1023 --      IF fShown & FocusOnSuperView THEN
18 1024 --1-      BEGIN
18 1025 --          aRect := thePort^.clipRgn^^.rgnBBox;
18 1026 --          QDToVRect(aRect, viewRect);
18 1027 --          {$H-}
18 1028 --          GetFrame(myFrame);
18 1029 --          IF SectVRect(myFrame, viewRect, visRect) THEN
18 1030 --          {$H+}
18 1031 --2-      BEGIN
18 1032 --          ViewToQDRect(myFrame, qdFrame); {Must be done before changing gLongOffset}
18 1033 --          FOR vhs := v TO h DO
18 1034 --              IF fSize.vh[vhs] > kMaxCoord THEN
18 1035 --3-      BEGIN
18 1036 --          relOrigin.vh[vhs] := viewRect.topLeft.vh[vhs] - visRect.topLeft.vh[vhs];
18 1037 --          gLongOffset.vh[vhs] := gLongOffset.vh[vhs] - fLocation.vh[vhs];
18 1038 --3-      END
18 1039 --          ELSE
18 1040 --3-      BEGIN
18 1041 --          relOrigin.vh[vhs] := -fLocation.vh[vhs] + gLongOffset.vh[vhs];
18 1042 --          gLongOffset.vh[vhs] := 0;
18 1043 --3-      END;
18 1044 --
18 1045 --          SetOrigin(thePort^.portRect.left + relOrigin.h,
18 1046 --                  thePort^.portRect.top + relOrigin.v);
18 1047 --          ClipFurtherTo(qdFrame, relOrigin.h, relOrigin.v);
18 1048 --
18 1049 --          gFocusedView := SELF;
18 1050 --          EXIT(Focus);
18 1051 --2-      END;
18 1052 --1-      END;
18 1053 --
18 1054 --      { If we get this far, then we can't focus }
18 1055 --      ClipRect(gZeroRect);
18 1056 --      gFocusedView := NIL;
18 1057 --      Focus := FALSE;
18 1058 --0 A  END (TView.Focus);
18 1059 --
18 1060 --      {-----+
18 1061 --      | FocusOnSuperView
18 1062 --      +-----}
18 1063 --      {$S Mares}
18 1064 -- A  FUNCTION TView.FocusOnSuperView: BOOLEAN;
18 1065 --0 A  BEGIN
18 1066 --      IF fSuperView <> NIL THEN
18 1067 --          FocusOnSuperView := fSuperView.Focus
18 1068 --      ELSE
18 1069 --          FocusOnSuperView := FALSE;
18 1070 --0 A  END (TView.FocusOnSuperView);
18 1071 --
18 1072 --      {-----+
18 1073 --      | ForceRedraw
18 1074 --      +-----}
18 1075 --      {$S Mares}
18 1076 -- A  PROCEDURE TView.ForceRedraw;
18 1077 --      VAR
18 1078 --          extent:    VRect;
18 1079 --
18 1080 --0 A  BEGIN
18 1081 --      GetExtent(extent);
18 1082 --      InvalidVRect(extent);
18 1083 --0 A  END (TView.ForceRedraw);
18 1084 --
18 1085 --      {-----+
18 1086 --      | Free
18 1087 --      +-----}
18 1088 --      {$S MAClipboard}

```

```

18 1089 -- A  PROCEDURE TView.FreeFromClipboard;
18 1090 --
18 1091 0- A  BEGIN
18 1092 --      IF fDocument <> NIL THEN
18 1093 --          fDocument.FreeFromClipboard
18 1094 --      ELSE
18 1095 --          Free;
18 1096 -0 A  END (TView.FreeFromClipboard);
18 1097 --
18 1098 --      { $IFC qInspector }
18 1099 --      {-----+
18 1100 --      |  GetInspectorName      |
18 1101 --      +-----+
18 1102 --      { $S MAINspector }
18 1103 -- A  PROCEDURE TView.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
18 1104 --      VAR
18 1105 --          itsWindow:    TWindow;
18 1106 --
18 1107 0- A  BEGIN
18 1108 --      IF IsObject(fSuperView) THEN
18 1109 1-      BEGIN
18 1110 --          itsWindow := GetWindow;
18 1111 --          IF IsObject(itsWindow) & (NOT ODD(ORD4(itsWindow.fWMgrWindow))) THEN
18 1112 2-      BEGIN
18 1113 --          GetWTitle(itsWindow.fWMgrWindow, inspectorName);
18 1114 --          inspectorName := CONCAT('In ', inspectorName);
18 1115 -2      END;
18 1116 -1      END;
18 1117 -0 A  END (TView.GetInspectorName);
18 1118 --      { $ENDC }
18 1119 --
18 1120 --      {-----+
18 1121 --      |  GetDialogView          |  Actually, returns a TDialogView..
18 1122 --      +-----+
18 1123 --      { $S MAREs }
18 1124 -- A  FUNCTION TView.GetDialogView: TView;
18 1125 0- A  BEGIN
18 1126 --      IF fSuperView <> NIL THEN
18 1127 --          GetDialogView := fSuperView.GetDialogView
18 1128 --      ELSE
18 1129 --          GetDialogView := NIL;
18 1130 -0 A  END (TView.GetDialogView);
18 1131 --
18 1132 --      {-----+
18 1133 --      |  GetExtent              |
18 1134 --      +-----+
18 1135 --      { $S MAREs }
18 1136 -- A  PROCEDURE TView.GetExtent (VAR itsExtent: VRect);
18 1137 0- A  BEGIN
18 1138 --          itsExtent.topLeft := gZeroVPt;
18 1139 --          itsExtent.botRight := fSize;
18 1140 -0 A  END (TView.GetExtent);
18 1141 --
18 1142 --      {-----+
18 1143 --      |  GetFrame                |
18 1144 --      +-----+
18 1145 --      { $S MAREs }
18 1146 -- A  PROCEDURE TView.GetFrame (VAR itsFrame: VRect);
18 1147 0- A  BEGIN
18 1148 --          itsFrame.topLeft := fLocation;
18 1149 --          itsFrame.right := itsFrame.left + fSize.h;
18 1150 --          itsFrame.bottom := itsFrame.top + fSize.v;
18 1151 -0 A  END (TView.GetFrame);
18 1152 --
18 1153 --      {-----+
18 1154 --      |  GetGrafPort              |
18 1155 --      +-----+
18 1156 --      { $S MAREs }
18 1157 -- A  FUNCTION TView.GetGrafPort: GrafPtr;
18 1158 0- A  BEGIN
18 1159 --      IF gPrinting THEN
18 1160 --          GetGrafPort := thePort      {thePort assumed to be set by print handler}
18 1161 --      ELSE IF fSuperView <> NIL THEN
18 1162 --          GetGrafPort := fSuperView.GetGrafPort
18 1163 --      ELSE
18 1164 --          GetGrafPort := NIL;
18 1165 -0 A  END (TView.GetGrafPort);
18 1166 --
18 1167 --      {-----+
18 1168 --      |  GetPrintExtent           |
18 1169 --      +-----+
18 1170 --      { $S MANonRes }
18 1171 -- A  PROCEDURE TView.GetPrintExtent (VAR printExtent: VRect);
18 1172 0- A  BEGIN
18 1173 --          GetExtent(printExtent);
18 1174 -0 A  END (TView.GetPrintExtent);
18 1175 --
18 1176 --      {-----+
18 1177 --      |  GetQDExtent              |
18 1178 --      +-----+
18 1179 --      { $S MAREs }
18 1180 -- A  PROCEDURE TView.GetQDExtent (VAR qdExtent: Rect);
18 1181 0- A  BEGIN
18 1182 --          qdExtent.topLeft := ViewToQDPt(gZeroVPt);
18 1183 --          qdExtent.botRight := ViewToQDPt(fSize);
18 1184 -0 A  END (TView.GetQDExtent);
18 1185 --
18 1186 --      {-----+
18 1187 --      |  GetScroller              |

```

```

18 1188 -- +-----+
18 1189 -- {$$ MRes}
18 1190 -- A FUNCTION TView.GetScroller (immediateSuperView: BOOLEAN): TScroller;
18 1191 --
18 1192 -- VAR
18 1193 --     aScroller:      TScroller;
18 1194 --
18 1195 0- A BEGIN
18 1196 --     GetScroller := NIL;
18 1197 --     IF fSuperView <> NIL THEN
18 1198 1- BEGIN
18 1199 --         aScroller := fSuperView.GetScroller(immediateSuperView);
18 1200 --         IF (aScroller = fSuperView) | (NOT immediateSuperView) THEN
18 1201 --             GetScroller := aScroller;
18 1202 -1 END;
18 1203 -0 A END {TView.GetScroller};
18 1204 --
18 1205 -- +-----+
18 1206 -- |   GetVisibleRect   |
18 1207 -- +-----+
18 1208 -- {$$ MRes}
18 1209 -- A PROCEDURE TView.GetVisibleRect (VAR visQDRect: Rect);
18 1210 --
18 1211 -- VAR
18 1212 --     extent:      VRect;
18 1213 --
18 1214 0- A BEGIN
18 1215 --     {!!! Need to make GetQDExtent work for scrollers}
18 1216 --     GetExtent(extent);
18 1217 --     ViewToQDRect(extent, visQDRect);
18 1218 --     VisibleRect(visQDRect);
18 1219 -0 A END {TView.GetVisibleRect};
18 1220 --
18 1221 -- +-----+
18 1222 -- |   GetWindow        |
18 1223 -- +-----+
18 1224 -- {$$ MRes}
18 1225 -- A FUNCTION TView.GetWindow: TWindow;
18 1226 0- A BEGIN
18 1227 --     IF fSuperView <> NIL THEN
18 1228 --         GetWindow := fSuperView.GetWindow
18 1229 --     ELSE
18 1230 --         GetWindow := NIL;
18 1231 -0 A END {TView.GetWindow};
18 1232 --
18 1233 -- +-----+
18 1234 -- |   GivePasteData    |
18 1235 -- +-----+
18 1236 -- {$$ MAClipboard}
18 1237 -- A FUNCTION TView.GivePasteData (aDataHandle: Handle; dataType: ResType): LONGINT;
18 1238 -- VAR
18 1239 --     offset:      LONGINT;
18 1240 --     err:         LONGINT;
18 1241 --     savedPerm:   BOOLEAN;
18 1242 --
18 1243 0- A BEGIN
18 1244 --     {Make sure the scrap handle is grown out of permanent memory. If it
18 1245 --     was grown out of temporary memory it would eat away at our
18 1246 --     code space.}
18 1247 --     savedPerm := PermAllocation(TRUE);
18 1248 --     err := GetScrap(aDataHandle, dataType, offset);
18 1249 --     savedPerm := PermAllocation(savedPerm);
18 1250 --
18 1251 --     {IFC qDebug}
18 1252 --     IF err < 0 THEN
18 1253 --         WriteLn('TView.GivePasteData got error code # ', err:1, ' from GetScrap for type "',
18 1254 --             dataType, '"');
18 1255 --     {SEND}
18 1256 --     GivePasteData := err;
18 1257 -0 A END {TView.GivePasteData};
18 1258 --
18 1259 -- +-----+
18 1260 -- |   HandleCursor     |
18 1261 -- +-----+
18 1262 -- {$$ MRes}
18 1263 -- A FUNCTION TView.HandleCursor (theMouse: VPoint; cursorRgn: RgnHandle): TView;
18 1264 --
18 1265 -- VAR
18 1266 --     dummyView,
18 1267 --     viewThatHandledCursor: TView;
18 1268 --
18 1269 -- B FUNCTION TestMouse (theSubView: TView): BOOLEAN;
18 1270 -- VAR
18 1271 --     subViewPt:      VPoint;
18 1272 --
18 1273 0- B BEGIN
18 1274 --     subViewPt := theMouse;
18 1275 --     theSubView.SuperToLocal(subViewPt);
18 1276 --     IF theSubView.ContainsMouse(subViewPt) THEN
18 1277 --         viewThatHandledCursor := theSubView.HandleCursor(subViewPt, cursorRgn);
18 1278 --     TestMouse := viewThatHandledCursor <> NIL;
18 1279 -0 B END {TestMouse};
18 1280 --
18 1281 0- A BEGIN
18 1282 --     {Assume if we got here then we already know the mouse is in this view}
18 1283 --     viewThatHandledCursor := NIL;
18 1284 --     dummyView := LastSubViewThat(TestMouse);
18 1285 --
18 1286 --     IF viewThatHandledCursor = NIL THEN

```

```

18 1287 --      IF IsViewEnabled & Focus & DoSetCursor(ViewToQDpt(theMouse), cursorRgn) THEN
18 1288 --          viewThatHandledCursor := SELF;
18 1289 --
18 1290 --      HandleCursor := viewThatHandledCursor;
18 1291 -- A  END {TView.HandleCursor};
18 1292 --
18 1293 --      {-----+
18 1294 --      |   HandleMouseDown
18 1295 --      +-----+}
18 1296 --      {$$ MASElCommand}
18 1297 -- A  FUNCTION TView.HandleMouseDown (theMouse:      VPoint;
18 1298 --                                VAR info:      EventInfo;
18 1299 --                                VAR hysteresis: Point;
18 1300 --                                VAR theCommand: TCommand): BOOLEAN;
18 1301 --
18 1302 --      VAR
18 1303 --          viewThatHandledMouse: TView;
18 1304 --
18 1305 --      {-----+
18 1306 --      |   TestMouse
18 1307 --      +-----+}
18 1307 -- B  FUNCTION TestMouse (theSubView: TView): BOOLEAN;
18 1308 --      VAR
18 1309 --          subViewPt:      VPoint;
18 1310 --
18 1311 --      BEGIN
18 1312 --          subViewPt := theMouse;
18 1313 --          theSubView.SuperToLocal(subViewPt);
18 1314 --          IF theSubView.ContainsMouse(subViewPt) THEN
18 1315 --              TestMouse := theSubView.HandleMouseDown(subViewPt,
18 1316 --                                                         info, hysteresis, theCommand)
18 1317 --          ELSE
18 1318 --              TestMouse := FALSE;
18 1319 --      END {TestMouse};
18 1320 --
18 1321 -- A  BEGIN
18 1322 --      HandleMouseDown := FALSE;
18 1323 --      theCommand := gNoChanges;
18 1324 --
18 1325 --      {Assume if we got here then we already know the mouse is in this view}
18 1326 --
18 1327 --      viewThatHandledMouse := LastSubviewThat(TestMouse);
18 1328 --
18 1329 --      IF viewThatHandledMouse <> NIL THEN      {a subview handled it}
18 1330 --          HandleMouseDown := TRUE
18 1331 --      ELSE IF IsViewEnabled & Focus THEN      {see if we can handle it}
18 1332 --          BEGIN
18 1333 --              theCommand := DoMouseCommand(ViewToQDpt(theMouse), info, hysteresis);
18 1334 --              HandleMouseDown := TRUE;
18 1335 --          END;
18 1336 --      END {TView.HandleMouseDown};
18 1337 --
18 1338 --      {-----+
18 1339 --      |   InvalidRect
18 1340 --      +-----+}
18 1341 --      {$$ MAREs}
18 1342 -- A  PROCEDURE TView.InvalidRect (r: Rect);
18 1343 --      BEGIN
18 1344 --          IF Focus THEN
18 1345 --              BEGIN
18 1346 --                  VisibleRect(r);
18 1347 --                  InvalRect(r);
18 1348 --              END;
18 1349 --      END {TView.InvalidRect};
18 1350 --
18 1351 --      {-----+
18 1352 --      |   InvalidVRect
18 1353 --      +-----+}
18 1354 --      {$$ MAREs}
18 1355 -- A  PROCEDURE TView.InvalidVRect (viewRect: VRect);
18 1356 --      VAR
18 1357 --          r:      Rect;
18 1358 --
18 1359 --      BEGIN
18 1360 --          IF Focus THEN
18 1361 --              BEGIN
18 1362 --                  ViewToQDRect(viewRect, r);
18 1363 --                  VisibleRect(r);
18 1364 --                  InvalRect(r);
18 1365 --              END;
18 1366 --      END {TView.InvalidVRect};
18 1367 --
18 1368 --      {-----+
18 1369 --      |   IsShown
18 1370 --      +-----+}
18 1371 --      {$$ MAREs}
18 1372 -- A  FUNCTION TView.IsShown: BOOLEAN;
18 1373 --      BEGIN
18 1374 --          IsShown := fShown;
18 1375 --      END;
18 1376 --
18 1377 --      {-----+
18 1378 --      |   IsViewEnabled
18 1379 --      +-----+}
18 1380 --      {$$ MAREs}
18 1381 -- A  FUNCTION TView.IsViewEnabled: BOOLEAN;
18 1382 --      BEGIN
18 1383 --          IsViewEnabled := fViewEnabled;
18 1384 --      END;
18 1385 --

```

```

18 1386 -- {-----+
18 1387 -- | LastSubviewThat |
18 1388 -- +-----+
18 1389 -- {$$ MRes}
18 1390 -- A FUNCTION TView.LastSubviewThat (FUNCTION TestSubview (theSubview: TView): BOOLEAN): TView;
18 1391 0- A BEGIN
18 1392 -- IF fSubviews <> NIL THEN
18 1393 --     LastSubviewThat := TView(fSubviews.LastThat(TestSubview))
18 1394 -- ELSE
18 1395 --     LastSubviewThat := NIL;
18 1396 0- A END (TView.LastSubviewThat);
18 1397 --
18 1398 -- {-----+
18 1399 -- | LocalToSuper |
18 1400 -- +-----+
18 1401 -- {$$ MRes}
18 1402 -- A PROCEDURE TView.LocalToSuper (VAR thePoint: VPoint);
18 1403 0- A BEGIN
18 1404 --     AdvPt(fLocation, thePoint);
18 1405 0- A END (TView.LocalToSuper);
18 1406 --
18 1407 -- {-----+
18 1408 -- | LocalToWindow |
18 1409 -- +-----+
18 1410 -- {$$ MRes}
18 1411 -- A PROCEDURE TView.LocalToWindow (VAR thePoint: VPoint);
18 1412 --
18 1413 -- VAR
18 1414 --     aView: TView;
18 1415 --
18 1416 0- A BEGIN
18 1417 --     aView := SELF;
18 1418 --     WHILE aView <> NIL DO
18 1419 1- BEGIN
18 1420 --         AdvPt(aView.fLocation, thePoint);
18 1421 --         aView := aView.fSuperview;
18 1422 1- END;
18 1423 0- A END (TView.LocalToWindow);
18 1424 --
18 1425 -- {-----+
18 1426 -- | Locate |
18 1427 -- +-----+
18 1428 -- {$$ MANonRes}
18 1429 -- A PROCEDURE TView.Locate (h, v: VCoordinate; invalidate: BOOLEAN);
18 1430 --
18 1431 -- {-----+
18 1432 -- | NotifySubview |
18 1433 -- +-----+
18 1434 -- B PROCEDURE NotifySubview (theSubview: TView);
18 1435 0- B BEGIN
18 1436 --     theSubview.SuperViewMoved(invalidate);
18 1437 0- B END;
18 1438 --
18 1439 0- A BEGIN
18 1440 --     IF (h <> fLocation.h) OR (v <> fLocation.v) THEN
18 1441 1- BEGIN
18 1442 --         IF invalidate THEN
18 1443 --             ForceRedraw;
18 1444 --             fLocation.h := h;
18 1445 --             fLocation.v := v;
18 1446 --             gFocusedView := NIL; {Must re-focus because view moved.}
18 1447 --             IF invalidate THEN
18 1448 --                 ForceRedraw;
18 1449 --
18 1450 --             IF fSuperview <> NIL THEN
18 1451 --                 fSuperview.SubViewMoved(SELF);
18 1452 --                 EachSubview(NotifySubview);
18 1453 1- END;
18 1454 0- A END (TView.Locate);
18 1455 --
18 1456 -- {-----+
18 1457 -- | MakeFirstSubview |
18 1458 -- +-----+
18 1459 -- {$$ MANonRes}
18 1460 -- A PROCEDURE TView.MakeFirstSubview (theSubview: TView);
18 1461 0- A BEGIN
18 1462 --     IF fSubviews <> NIL THEN {there must be at least one subview}
18 1463 --         IF TView(fSubviews.First) <> theSubview THEN
18 1464 1- BEGIN
18 1465 --             fSubviews.Delete(theSubview);
18 1466 --             fSubviews.InsertFirst(theSubview);
18 1467 --             theSubview.ForceRedraw;
18 1468 1- END;
18 1469 0- A END (TView.MakeFirstSubview);
18 1470 --
18 1471 -- {-----+
18 1472 -- | MakeLastSubview |
18 1473 -- +-----+
18 1474 -- {$$ MANonRes}
18 1475 -- A PROCEDURE TView.MakeLastSubview (theSubview: TView);
18 1476 0- A BEGIN
18 1477 --     IF fSubviews <> NIL THEN {there must be at least one subview}
18 1478 --         IF TView(fSubviews.Last) <> theSubview THEN
18 1479 1- BEGIN
18 1480 --             fSubviews.Delete(theSubview);
18 1481 --             fSubviews.InsertLast(theSubview);
18 1482 --             theSubview.ForceRedraw;
18 1483 1- END;
18 1484 0- A END (TView.MakeLastSubview);

```



```

18 1485 --
18 1486 -- {-----+
18 1487 -- |   Open   |
18 1488 -- +-----+
18 1489 -- {$$ MAOpen}
18 1490 -- A  PROCEDURE TView.Open;
18 1491 --
18 1492 -- {-----+
18 1493 -- |   OpenSubView   |
18 1494 -- +-----+
18 1495 -- B  PROCEDURE OpenSubView (theSubView: TView);
18 1496 0- B  BEGIN
18 1497 --      theSubView.Open;
18 1498 0- B  END;
18 1499 --
18 1500 0- A  BEGIN
18 1501 --      EachSubView(OpenSubView);
18 1502 0- A  END (TView.Open);
18 1503 --
18 1504 -- {-----+
18 1505 -- |   PageInteriorChanged   |
18 1506 -- +-----+
18 1507 -- {$$ MANonRes}
18 1508 -- A  PROCEDURE TView.PageInteriorChanged (newInterior: Rect);
18 1509 0- A  BEGIN
18 1510 0- A  END (TView.PageInteriorChanged);
18 1511 --
18 1512 -- {-----+
18 1513 -- |   QDToViewPt   |
18 1514 -- +-----+
18 1515 -- {$$ MAREs}
18 1516 -- A  PROCEDURE TView.QDToViewPt (qdPoint: Point; VAR viewPt: VPoint);
18 1517 0- A  BEGIN
18 1518 --      PtToVpt(qdPoint, viewPt);
18 1519 --      AddVpt(gLongOffset, viewPt);
18 1520 0- A  END (TView.QDToViewPt);
18 1521 --
18 1522 -- {-----+
18 1523 -- |   QDToViewRect   |
18 1524 -- +-----+
18 1525 -- {$$ MAREs}
18 1526 -- A  PROCEDURE TView.QDToViewRect (qdRect: Rect; VAR viewRect: VRect);
18 1527 0- A  BEGIN
18 1528 --      RectToVrect(qdRect, viewRect);
18 1529 --      OffsetVrect(viewRect, gLongOffset.h, gLongOffset.v);
18 1530 0- A  END (TView.QDToViewRect);
18 1531 --
18 1532 -- {-----+
18 1533 -- |   RemoveSubView   |
18 1534 -- +-----+
18 1535 -- {$$ MANonRes}
18 1536 -- A  PROCEDURE TView.RemoveSubView (theSubView: TView);
18 1537 0- A  BEGIN
18 1538 --      IF fSubViews <> NIL THEN
18 1539 --          fSubViews.Delete(theSubView);
18 1540 --
18 1541 --      IF theSubView.fNextHandler = SELF THEN      { If the next event handler is the superview }
18 1542 --          theSubView.fNextHandler := NIL;        { take it out of the event handler chain. }
18 1543 --
18 1544 --      theSubView.fSuperView := NIL;
18 1545 0- A  END (TView.RemoveSubView);
18 1546 --
18 1547 -- {-----+
18 1548 -- |   Resize   |
18 1549 -- +-----+
18 1550 -- {$$ MANonRes}
18 1551 -- A  PROCEDURE TView.Resize (width, height: VCoordinate; invalidate: BOOLEAN);
18 1552 --
18 1553 -- VAR
18 1554 --     delta:      VPoint;
18 1555 --     aRgn:      RgnHandle;
18 1556 --     myFrame:    VRect;
18 1557 --     r:          Rect;
18 1558 --
18 1559 -- {-----+
18 1560 -- |   NotifySubView   |
18 1561 -- +-----+
18 1562 -- B  PROCEDURE NotifySubView (theSubView: TView);
18 1563 0- B  BEGIN
18 1564 --      theSubView.SuperViewChangedSize(delta, invalidate);
18 1565 0- B  END;
18 1566 --
18 1567 0- A  BEGIN
18 1568 --      IF (fSize.h <> width ) OR (fSize.v <> height) THEN
18 1569 1-  BEGIN
18 1570 --          SetVpt(delta, width - fSize.h, height - fSize.v);
18 1571 --          GetFrame(myFrame);          { Get the old frame, before we change the size }
18 1572 --          fSize.h := width;
18 1573 --          fSize.v := height;
18 1574 --          qFocusedView := NIL;        {Must re-focus because size changed}
18 1575 --
18 1576 --          IF invalidate & FocusOnSuperView THEN
18 1577 2-  BEGIN
18 1578 --              {!!! Could be sped up by reusing a region rather than creating one}
18 1579 --              aRgn := MakeNewRgn;
18 1580 --              ViewToQDRect(myFrame, r);
18 1581 --              VisibleRect(r);
18 1582 --              RectRgn(aRgn, r);
18 1583 --

```

```

18 1584 --      GetFrame(myFrame);          { Now get the new frame }
18 1585 --      ViewToQDRect(myFrame, r);
18 1586 --      VisibleRect(r);
18 1587 --      { $IFC qDebug }
18 1588 --      UseTempRgn('TView.Resize');
18 1589 --      { $ENDC }
18 1590 --      RectRgn(gTempRgn, r);
18 1591 --
18 1592 --      XOrRgn(aRgn, gTempRgn, gTempRgn);
18 1593 --      InvalRgn(gTempRgn);
18 1594 --      { $IFC qDebug }
18 1595 --      DoneWithTempRgn;
18 1596 --      { $ENDC }
18 1597 --      DisposeRgn(aRgn);
18 1598 --      END;
18 1599 --
18 1600 --      IF fSuperview <> NIL THEN
18 1601 --          fSuperview.SubViewChangedSize(SELF, delta);
18 1602 --          EachSubView(NotifySubView);
18 1603 --      END;
18 1604 -- 0 A  END {TView.Resize};
18 1605 --
18 1606 --      {-----+
18 1607 --      |   RevealBottom   |
18 1608 --      +-----+
18 1609 --      { $S MASCROLL }
18 1610 -- 0 A  PROCEDURE TView.RevealBottom (redraw: BOOLEAN);
18 1611 --
18 1612 --      VAR
18 1613 --          rectToReveal:   VRect;
18 1614 --
18 1615 -- 0- A  BEGIN
18 1616 --          rectToReveal.topLeft := fSize;
18 1617 --          rectToReveal.botRight := fSize;
18 1618 --          RevealRect(rectToReveal, Point(0), redraw);
18 1619 -- 0 A  END {TView.RevealBottom};
18 1620 --
18 1621 --      {-----+
18 1622 --      |   RevealRect    |
18 1623 --      +-----+
18 1624 --      { $S MASCROLL }
18 1625 -- 0 A  PROCEDURE TView.RevealRect (rectToReveal: VRect; minToSee: Point;
18 1626 --                                redraw: BOOLEAN);
18 1627 -- 0- A  BEGIN
18 1628 --          IF fSuperview <> NIL THEN
18 1629 --              BEGIN
18 1630 --                  OffsetVRect(rectToReveal, fLocation.h, fLocation.v);
18 1631 --                  fSuperview.RevealRect(rectToReveal, minToSee, redraw);
18 1632 --              END;
18 1633 -- 0 A  END {TView.RevealRect};
18 1634 --
18 1635 --      {-----+
18 1636 --      |   RevealTop     |
18 1637 --      +-----+
18 1638 --      { $S MASCROLL }
18 1639 -- 0 A  PROCEDURE TView.RevealTop (redraw: BOOLEAN);
18 1640 -- 0- A  BEGIN
18 1641 --          RevealRect(gZeroVRect, Point(0), redraw);
18 1642 -- 0 A  END {TView.RevealTop};
18 1643 --
18 1644 --      {-----+
18 1645 --      |   Show          |
18 1646 --      +-----+
18 1647 --      { $S MANonRes }
18 1648 -- 0 A  PROCEDURE TView.Show (state, redraw: BOOLEAN);
18 1649 -- 0- A  BEGIN
18 1650 --          redraw := redraw AND (state <> fShown);
18 1651 --          IF redraw THEN
18 1652 --              BEGIN
18 1653 --                  fShown := TRUE;
18 1654 --                  ForceRedraw;
18 1655 --              END;
18 1656 --          fShown := state;
18 1657 -- 0 A  END;
18 1658 --
18 1659 --      {-----+
18 1660 --      |   ShowReverted  |
18 1661 --      +-----+
18 1662 --      { $S MAREadFile }
18 1663 -- 0 A  PROCEDURE TView.ShowReverted;
18 1664 --
18 1665 --          {-----+
18 1666 --          |   RevertSubView   |
18 1667 --          +-----+
18 1668 -- 0 B  PROCEDURE RevertSubView (theSubView: TView);
18 1669 -- 0- B  BEGIN
18 1670 --          theSubView.ShowReverted;
18 1671 -- 0 B  END;
18 1672 --
18 1673 -- 0- A  BEGIN
18 1674 --          AdjustSize;
18 1675 --          ForceRedraw;
18 1676 --          EachSubView(RevertSubView);
18 1677 -- 0 A  END {TView.ShowReverted};
18 1678 --
18 1679 --      {-----+
18 1680 --      |   SubViewChangedSize   |
18 1681 --      +-----+
18 1682 --      { $S MANonRes }

```

```

18 1683 -- A  PROCEDURE TView.SubViewChangedSize (theSubView: TView; delta: VPoint);
18 1684 0- A  BEGIN
18 1685 -0 A  END (TView.SubViewChangedSize);
18 1686 --
18 1687 --  {-----+
18 1688 --  | SubViewMoved |
18 1689 --  +-----}
18 1690 --  {$S MANonRes}
18 1691 -- A  PROCEDURE TView.SubViewMoved (theSubView: TView);
18 1692 0- A  BEGIN
18 1693 -0 A  END (TView.SubViewMoved);
18 1694 --
18 1695 --  {-----+
18 1696 --  | SuperToLocal |
18 1697 --  +-----}
18 1698 --  {$S MAREs}
18 1699 -- A  PROCEDURE TView.SuperToLocal (VAR thePoint: VPoint);
18 1700 0- A  BEGIN
18 1701 --      SubVpt(fLocation, thePoint);
18 1702 -0 A  END (TView.SuperToLocal);
18 1703 --
18 1704 --  {-----+
18 1705 --  | SuperViewChangedSize |
18 1706 --  +-----}
18 1707 --  {$S MANonRes}
18 1708 -- A  PROCEDURE TView.SuperViewChangedSize (delta: VPoint; invalidate: BOOLEAN);
18 1709 --  VAR
18 1710 --      newSize:      VPoint;
18 1711 --      needsResizing: BOOLEAN;
18 1712 --      vhs:          VHSelect;
18 1713 --
18 1714 0- A  BEGIN
18 1715 --      needsResizing := FALSE;
18 1716 --      newSize := fSize;
18 1717 --      FOR vhs := v TO h DO
18 1718 --          IF fSizeDeterminer[vhs] = sizeSuperView THEN
18 1719 --              needsResizing := TRUE
18 1720 --          ELSE IF fSizeDeterminer[vhs] = sizeRelSuperView THEN
18 1721 1- BEGIN
18 1722 --              newSize.vh[vhs] := newSize.vh[vhs] + delta.vh[vhs];
18 1723 --              needsResizing := TRUE;
18 1724 -1 END;
18 1725 --
18 1726 --      IF needsResizing THEN
18 1727 1- BEGIN
18 1728 --          ComputeSize(newSize);
18 1729 --          Resize(newSize.h, newSize.v, invalidate);
18 1730 --          DoPagination;
18 1731 -1 END;
18 1732 -0 A  END (TView.SuperViewChangedSize);
18 1733 --
18 1734 --  {-----+
18 1735 --  | SuperViewMoved |
18 1736 --  +-----}
18 1737 --  {$S MANonRes}
18 1738 -- A  PROCEDURE TView.SuperViewMoved (invalidate: BOOLEAN);
18 1739 0- A  BEGIN
18 1740 -0 A  END (TView.SuperViewMoved);
18 1741 --
18 1742 --  {-----+
18 1743 --  | Update |
18 1744 --  +-----}
18 1745 --  {$S MAREs}
18 1746 -- A  PROCEDURE TView.Update;
18 1747 --
18 1748 --  VAR
18 1749 --      itsWindow:      TWindow;
18 1750 --
18 1751 0- A  BEGIN
18 1752 --      itsWindow := GetWindow;
18 1753 --      IF itsWindow <> NIL THEN
18 1754 1- BEGIN
18 1755 --          BeginUpdate(itsWindow.fWMgrWindow);
18 1756 --          DrawContents;
18 1757 --          EndUpdate(itsWindow.fWMgrWindow);
18 1758 -1 END;
18 1759 -0 A  END (TView.Update);
18 1760 --
18 1761 --  {-----+
18 1762 --  | ValidVRect |
18 1763 --  +-----}
18 1764 --  {$S MAREs}
18 1765 -- A  PROCEDURE TView.ValidVRect (viewRect: VRect);
18 1766 --  VAR
18 1767 --      r:      Rect;
18 1768 --
18 1769 0- A  BEGIN
18 1770 --      IF Focus THEN
18 1771 1- BEGIN
18 1772 --          ViewToQDRect(viewRect, r);
18 1773 --          VisibleRect(r);
18 1774 --          ValidRect(r);
18 1775 -1 END;
18 1776 -0 A  END (TView.ValidVRect);
18 1777 --
18 1778 --  {-----+
18 1779 --  | ViewEnable |
18 1780 --  +-----}
18 1781 --  {$S MAREs}

```

```

18 1782 -- A  PROCEDURE TView.ViewEnable (state, redraw: BOOLEAN);
18 1783 0- A  BEGIN
18 1784 --      fViewEnabled := state;
18 1785 --      IF redraw THEN
18 1786 --          ForceRedraw;
18 1787 -0 A  END;
18 1788 --
18 1789 --      {-----+
18 1790 --      |   ViewToQDPt   |
18 1791 --      +-----}
18 1792 --      {$S MArEs}
18 1793 -- A  FUNCTION TView.ViewToQDPt (viewPt: VPoint): Point;
18 1794 0- A  BEGIN
18 1795 --      SubVPt(gLongOffset, viewPt);
18 1796 --      ViewToQDPt := VPtToPt(viewPt);
18 1797 -0 A  END (TView.ViewToQDPt);
18 1798 --
18 1799 --      {-----+
18 1800 --      |   ViewToQDRect  |
18 1801 --      +-----}
18 1802 --      {$S MArEs}
18 1803 -- A  PROCEDURE TView.ViewToQDRect (viewRect: VRect; VAR qdRect: Rect);
18 1804 0- A  BEGIN
18 1805 --      OffsetVRect(viewRect, -gLongOffset.h, -gLongOffset.v);
18 1806 --      VRectToRect(viewRect, qdRect);
18 1807 -0 A  END (TView.ViewToQDRect);
18 1808 --
18 1809 --      {-----+
18 1810 --      |   WindowToLocal  |
18 1811 --      +-----}
18 1812 --      {$S MArEs}
18 1813 -- A  PROCEDURE TView.WindowToLocal (VAR thePoint: VPoint);
18 1814 --
18 1815 --      VAR
18 1816 --          aView:    TView;
18 1817 --
18 1818 0- A  BEGIN
18 1819 --          aView := SELF;
18 1820 --          WHILE aView <> NIL DO
18 1821 1-              BEGIN
18 1822 --                  SubVPt(aView.fLocation, thePoint);
18 1823 --                  aView := aView.fSuperview;
18 1824 -1              END;
18 1825 -0 A  END (TView.WindowToLocal);
18 1826 --
18 1827 --      {-----+
18 1828 --      |   WriteToDeskScrap  |
18 1829 --      +-----}
18 1830 --      {$S MANonRes}
18 1831 -- A  PROCEDURE TView.WriteToDeskScrap;
18 1832 --      VAR
18 1833 --          pHndl:    PicHandle;
18 1834 --          qdExtent: Rect;
18 1835 --          err:      LONGINT;
18 1836 --          tempPort: GrafPort;
18 1837 --          tempCPort: CGrafPort;
18 1838 --
18 1839 0- A  BEGIN
18 1840 --          GetQDExtent(qdExtent);
18 1841 --
18 1842 --          IF gConfiguration.hasColorQD THEN
18 1843 --              OpenCPort(@tempCPort)
18 1844 --          ELSE
18 1845 --              OpenPort(@tempPort);
18 1846 --
18 1847 --          {Open color or black & white, depending on the port}
18 1848 --          pHndl := OpenPicture(qdExtent);
18 1849 --
18 1850 --          IF pHndl <> NIL THEN
18 1851 1-              BEGIN
18 1852 --
18 1853 --                  gDrawingPictScrap := TRUE;
18 1854 --                  ClipRect(qdExtent);
18 1855 --
18 1856 --                  Draw(qdExtent);
18 1857 --
18 1858 --                  gDrawingPictScrap := FALSE;
18 1859 --                  ClosePicture;
18 1860 --
18 1861 --                  {On the 128K ROMs the picFrame will be empty if drawing the
18 1862 --                  picture failed. On the 64K ROM's QuickDraw simply bombs.}
18 1863 --                  IF EmptyRect(pHndl^.picFrame) THEN
18 1864 2-                      BEGIN
18 1865 --                      {$IFC qDebug}
18 1866 --                          WRITELN('Picture frame is empty!');
18 1867 --                      {$ENDC}
18 1868 --                          KillPicture(pHndl);
18 1869 --                          Failure(memFullErr, 0);
18 1870 -2                      END;
18 1871 --
18 1872 --                      err := PutDeskScrapData('PICT', Handle(pHndl));
18 1873 --                      KillPicture(pHndl);
18 1874 --
18 1875 --                      IF err <> NoErr THEN
18 1876 2-                          BEGIN
18 1877 --                          {$IFC qDebug}
18 1878 --                              ProgramBreak('Failed to put PICT-type scrap');
18 1879 --                          {$ENDC}
18 1880 --                              Failure(err, 0);

```

```

18 1881 --      END;
18 1882 --      END
18 1883 --      ELSE
18 1884 --      BEGIN
18 1885 --      ($IFC qDebug)
18 1886 --      ProgramBreak('Couldn't OpenPicture during attempt to write PICT to desk scrap');
18 1887 --      ($ENDC)
18 1888 --      Failure(memFullErr, 0); {Assume cause of failure was lack of memory}
18 1889 --      END;
18 1890 --
18 1891 --      IF gConfiguration.hasColorQD THEN
18 1892 --      CloseCPort(@tempCPort)
18 1893 --      ELSE
18 1894 --      ClosePort(@tempPort);
18 1895 -- A  END {TView.WriteToDeskScrap};
18 1896 --
18 1897 --      ($IFC qDebug)
18 1898 --      {-----+
18 1899 --      | Fields                                     |
18 1900 --      +-----}
18 1901 --      {$S MAFields}
18 1902 -- A  PROCEDURE TView.Fields (PROCEDURE DoToField (fieldName: Str255;
18 1903 --      fieldAddr: Ptr;
18 1904 --      fieldType: INTEGER)); OVERRIDE;
18 1905 -- A  BEGIN
18 1906 --      DoToField('TView', NIL, bClass);
18 1907 --      DoToField('fSuperView', @fSuperView, bObject);
18 1908 --      DoToField('fSubViews', @fSubViews, bObject);
18 1909 --      DoToField('fDocument', @fDocument, bObject);
18 1910 --      DoToField('fLocation', @fLocation, bVPoint);
18 1911 --      DoToField('fSize', @fSize, bVPoint);
18 1912 --      DoToField('fSizeDeter[h]', @fSizeDeterminer[h], bByte);
18 1913 --      DoToField('fSizeDeter[v]', @fSizeDeterminer[v], bByte);
18 1914 --      DoToField('fHLDdesired', @fHLDdesired, bByte);
18 1915 --      DoToField('fIdentifier', @fIdentifier, bOSType);
18 1916 --      DoToField('fShown', @fShown, bBoolean);
18 1917 --      DoToField('fViewEnabled', @fViewEnabled, bBoolean);
18 1918 --      DoToField('fPrintHandler', @fPrintHandler, bObject);
18 1919 --      INHERITED Fields(DoToField);
18 1920 -- A  END {TView.Fields};
18 1921 --      ($ENDC)
13 3006 --      {$I UMacApp.TWindow.p}

```

```

19 1 -- { $P }
19 2 -- { UMacApp.TWindow.p }
19 3 -- { Copyright © 1987, 1988 by Apple Computer Inc. All rights reserved. }
19 4 --
19 5 --
19 6 -- (*****
19 7 -- (*      T W i n d o w      *)
19 8 -- (*****
19 9 --
19 10 -- {$IFC qProceduralViews}
19 11 -- {$S MAOpen}
19 12 -- {-----+
19 13 -- | IWindow |
19 14 -- +-----}
19 15 -- A PROCEDURE TWindow.IWindow (itsDocument: TDocument; itsWmgrWindow: WindowPtr;
19 16 --                               canResize, canClose, disposeOnFree: BOOLEAN);
19 17 --
19 18 -- VAR
19 19 --     preDocname:      INTEGER;
19 20 --     constTitle:     INTEGER;
19 21 --     aString:         Str255;
19 22 --     fi:              FailInfo;
19 23 --
19 24 --     {-----+
19 25 --     | HandleFailure |
19 26 --     +-----}
19 27 -- B PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
19 28 -- 0- B BEGIN
19 29 --     Free;
19 30 -- 0 B END;
19 31 --
19 32 -- 0- A BEGIN
19 33 --     { set up fields necessary to free myself }
19 34 --     fWmgrWindow := itsWmgrWindow;
19 35 --     fFreeOnClosing := FALSE;
19 36 --     fDisposeOnFree := disposeOnFree;
19 37 --     fDocument := NIL;
19 38 --
19 39 --     SetPort(itsWmgrWindow); { ??? Why is this necessary? }
19 40 --     { Start with size of (0, 0). Will be corrected in TWindow.Open }
19 41 --     IView(NIL, NIL, gZeroVpt, gZeroVpt, SizeVariable, SizeVariable);
19 42 --
19 43 --     fProcID := GetWRefCon(itsWmgrWindow);
19 44 --     SetWRefCon(itsWmgrWindow, LONGINT(SELF));
19 45 --
19 46 --     fAdapted := FALSE; { Set by AdaptToScreen }
19 47 --     fHorzCentered := FALSE; { Set by Center }
19 48 --     fVertCentered := FALSE; { Set by Center }
19 49 --     fStaggered := FALSE; { Set by SimpleStagger }
19 50 --     fForcedOnScreen := FALSE; { Set by ForceOnScreen }
19 51 --     fIsActive := FALSE;
19 52 --     fIsResizable := canResize;
19 53 --     fIsClosable := canClose;
19 54 --     fTarget := SELF;
19 55 --     fTargetID := fIdentifier;
19 56 --     fClosesDocument := TRUE;
19 57 --     fOpenInitially := TRUE;
19 58 --     fIsModal := FALSE;
19 59 --     fDoFirstClick := FALSE;
19 60 --
19 61 --     CatchFailures(fi, HandleFailure);
19 62 --
19 63 --     fMoveBounds := gStdWMoveBounds;
19 64 --     SetResizeLimits(gStdWSizeRect.topLeft, gStdWSizeRect.botRight);
19 65 --     GetWTitle(itsWmgrWindow, aString);
19 66 --     IF ParseTitleTemplate(aString, preDocname, constTitle) THEN
19 67 --         SetWTitle(itsWmgrWindow, aString);
19 68 --     fPreDocname := preDocname;
19 69 --     fConstTitle := constTitle;
19 70 --
19 71 --     InstallDocument(itsDocument);
19 72 --
19 73 --     Success(fi);
19 74 -- 0 A END { TWindow.IWindow };
19 75 -- {SENDC}
19 76 --
19 77 --
19 78 -- {$IFC qTemplateViews}
19 79 -- {$S MAOpen}
19 80 -- {-----+
19 81 -- | IRes |
19 82 -- +-----}
19 83 -- A PROCEDURE TWindow.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr);
19 84 --
19 85 -- VAR
19 86 --     bounds:      Rect;
19 87 --     aWmgrWindow: WindowPtr;
19 88 --     preDocname:   INTEGER;
19 89 --     constTitle:   INTEGER;
19 90 --     aString:      Str255;
19 91 --     fi:           FailInfo;
19 92 --
19 93 --     {-----+
19 94 --     | HandleFailure |
19 95 --     +-----}
19 96 -- B PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
19 97 -- 0- B BEGIN
19 98 --     Free;

```

```

19 99 -0 B      END;
19 100 --
19 101 0- A      BEGIN
19 102 --          { set up fields necessary to free myself }
19 103 --          fDocument := NIL;
19 104 --          fWmgrWindow := NIL;
19 105 --          fFreeOnClosing := FALSE;
19 106 --          fDisposeOnFree := TRUE;
19 107 --
19 108 --          INHERITED IRes(NIL, itsSuperView, itsParams);
19 109 --
19 110 --          SetRect(bounds, fLocation.h, fLocation.v, fLocation.h + fSize.h, fLocation.v + fSize.v);
19 111 --          fLocation := gZeroVPT;
19 112 --
19 113 --          WITH WindowTemplatePtr(itsParams)^DO
19 114 1-          BEGIN
19 115 --              CatchFailures(fi, HandleFailure);
19 116 --
19 117 --              fProcID := procID;
19 118 --
19 119 --              IF gConfiguration.hasColorQD THEN
19 120 --                  aWmgrWindow := WindowPtr(NewCWindow(NIL, bounds, title, FALSE, procID, Pointer(-1),
19 121 --                  .                  hasGoAway, ORD4(SELF)))
19 122 --              ELSE
19 123 --                  aWmgrWindow := NewWindow(NIL, bounds, title, FALSE, procID, Pointer(-1),
19 124 --                  .                  hasGoAway, ORD4(SELF));
19 125 --              fWmgrWindow := aWmgrWindow;
19 126 --
19 127 --              fIsActive := FALSE;
19 128 --              fIsResizable := resizable;
19 129 --              fIsClosable := hasGoAway;
19 130 --              fTarget := SELF;
19 131 --              fTargetID := targetID;
19 132 --              fIsModal := isModal;
19 133 --              fDoFirstClick := doFirstClick;
19 134 --              fFreeOnClosing := freeOnClosing;
19 135 --              fDisposeOnFree := disposeOnFree;
19 136 --              fClosesDocument := closesDocument;
19 137 --              fOpenInitially := openInitially;
19 138 --              fMoveBounds := gStdWMoveBounds;
19 139 --              SetResizeLimits(gStdWSizeRect.topLeft, gStdWSizeRect.botRight);
19 140 --
19 141 --              GetWTitle(aWmgrWindow, aString);
19 142 --              IF ParseTitleTemplate(aString, preDocname, constTitle) THEN
19 143 --                  SetWTitle(aWmgrWindow, aString);
19 144 --                  fPreDocname := preDocname;
19 145 --                  fConstTitle := constTitle;
19 146 --
19 147 --              InstallDocument(itsDocument);
19 148 --
19 149 --              Success(fi);
19 150 --
19 151 --              IF mustAdaptToScreen THEN
19 152 --                  AdaptToScreen;
19 153 --              IF horzCenter | vertCenter THEN
19 154 --                  Center(horzCenter, vertCenter);
19 155 --              IF stagger THEN
19 156 --                  SimpleStagger(kStdStaggerAmount, kStdStaggerAmount, gStdStaggerCount);
19 157 --              IF mustForceOnScreen THEN
19 158 --                  ForceOnScreen;
19 159 --
19 160 --              OffsetPtrWStr(itsParams, SIZEOF(WindowTemplate));
19 161 -1          END;
19 162 -0 A      END { TWindow.IRes };
19 163 --          {$ENDC}
19 164 --
19 165 --
19 166 --          {$IFC qWriteTemplates}
19 167 --          {$S MAWriteRes}
19 168 --          {-----+
19 169 --          | WRes
19 170 --          +-----}
19 171 - A      PROCEDURE TWindow.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE:
19 172 --
19 173 --          VAR
19 174 --              theTitle: Str255;
19 175 --              wnPtr: WindowTemplatePtr;
19 176 --
19 177 0- A      BEGIN
19 178 --          INHERITED WRes(theResource, itsParams);
19 179 --
19 180 --          GetWTitle(fWmgrWindow, theTitle);
19 181 --
19 182 --          wnPtr := WindowTemplatePtr(ExpandPtrWStr(theResource, itsParams,
19 183 --          .          SIZEOF(WindowTemplate), LENGTH(theTitle)));
19 184 --
19 185 --          WITH wnPtr^ DO
19 186 1-          BEGIN
19 187 --              procID := fProcID;
19 188 --              hasGoAway := fIsClosable;
19 189 --              resizable := fIsResizable;
19 190 --              isModal := fIsModal;
19 191 --              doFirstClick := fDoFirstClick;
19 192 --              freeOnClosing := fFreeOnClosing;
19 193 --              disposeOnFree := fDisposeOnFree;
19 194 --              closesDocument := fClosesDocument;
19 195 --              openInitially := fOpenInitially;
19 196 --              mustAdaptToScreen := fAdapted;
19 197 --              stagger := fStaggered;

```

```

19 198 --      mustForceOnScreen := fForcedOnScreen;
19 199 --      vertCenter := fVertCentered;
19 200 --      horzCenter := fHorzCentered;
19 201 --      targetID := fTargetID;
19 202 --      (* title := theTitle; *)
19 203 --      CopyStr255(theTitle, PRStr(title));
19 204 -1      END;
19 205 -0 A  END;
19 206 --      {$ENDC}
19 207 --
19 208 --
19 209 --      {$IFC qWriteTemplates}
19 210 --      {$S MAWriteRes}
19 211 --      {-----+
19 212 --      |   WriteRes   |
19 213 --      +-----+}
19 214 -- A  PROCEDURE TWindow.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
19 215 0- A  BEGIN
19 216 --      gWResSignature := 'wind';  gWResType := 'TWindow';
19 217 --      WRes(theResource, itsParams);
19 218 -0 A  END;
19 219 --      {$ENDC}
19 220 --
19 221 --
19 222 --      {$S MAClose}
19 223 --      {-----+
19 224 --      |   Free   |
19 225 --      +-----+}
19 226 -- A  PROCEDURE TWindow.Free; OVERRIDE;
19 227 --
19 228 --      VAR
19 229 --      disposeOnFree: BOOLEAN;
19 230 --      wmgrWindow: WindowPtr;
19 231 --
19 232 0- A  BEGIN
19 233 --      disposeOnFree := fDisposeOnFree;
19 234 --      wmgrWindow := fWmgrWindow;
19 235 --
19 236 --      IF fDocument <> NIL THEN
19 237 --      fDocument.DeleteWindow(SELf);
19 238 --      ELSE
19 239 --      gApplication.DeleteFreeWindow(SELf);
19 240 --
19 241 --      INHERITED Free;
19 242 --
19 243 --      { ***** DON'T REFER TO ANY FIELDS OR METHODS OF SELF ***** }
19 244 --
19 245 --      FreeWmgrWindow(wmgrWindow, disposeOnFree);
19 246 -0 A  END { TWindow.Free };
19 247 --
19 248 --
19 249 --      {$S MAActivate}
19 250 --      {-----+
19 251 --      |   Activate   |
19 252 --      +-----+}
19 253 -- A  PROCEDURE TWindow.Activate (entering: BOOLEAN);
19 254 --
19 255 --      VAR
19 256 --      different: BOOLEAN;
19 257 --
19 258 0- A  BEGIN
19 259 --      gFocusedView := NIL; { Must refocus to make sure thePort is set }
19 260 --      different := entering <> fIsActive;
19 261 --
19 262 --      IF different THEN
19 263 1-      BEGIN
19 264 --      Update;
19 265 --      INHERITED Activate(entering);
19 266 --
19 267 --      IF entering THEN
19 268 2-      BEGIN
19 269 --      IF gFrontWindow <> NIL THEN { So lost deactivate doesn't
19 270 --      screw up gTarget }
19 271 --      gFrontWindow.Activate(FALSE);
19 272 --      gFrontWindow := SELF;
19 273 --      gApplication.SetTarget(fTarget);
19 274 --      gDocument := fDocument;
19 275 -2      END
19 276 --      ELSE
19 277 2-      BEGIN
19 278 --      fTarget := gTarget;
19 279 --      gApplication.SetTarget(gApplication);
19 280 --      gFrontWindow := NIL;
19 281 --      gDocument := NIL;
19 282 --      SetCursor(arrow); { ensure that the cursor is an arrow
19 283 --      when MacApp loses control }
19 284 -2      END;
19 285 -1      END;
19 286 --
19 287 --      IF fIsResizable THEN
19 288 --      DrawResizeIcon;
19 289 --
19 290 --      fIsActive := entering;
19 291 -0 A  END { TWindow.Activate };
19 292 --
19 293 --
19 294 --      {$S MAOpen}
19 295 --      {-----+
19 296 --      |   AdaptToScreen   |

```



```

19 297 -- +-----)
19 298 -- A  PROCEDURE TWindow.AdaptToScreen;
19 299 --
19 300 --  CONST
19 301 --      stdHScreen = 512;
19 302 --      stdVScreen = 342;
19 303 --      stdScreen = $10000 * stdVScreen + stdHScreen;
19 304 --
19 305 --  VAR
19 306 --      diff:      Point;
19 307 --      windRect:  Rect;
19 308 --      newSize:   Point;
19 309 --
19 310 0- A  BEGIN
19 311 --      fAdapted := TRUE;           { We adapted to the screen }
19 312 --      { Compute difference between current screen and std Mac screen }
19 313 --      diff := screenBits.bounds.botRight;
19 314 --      SubPt(screenBits.bounds.topLeft, diff);
19 315 --      SubPt(Point(stdScreen), diff);
19 316 --
19 317 --      IF LONGINT(diff) <> 0 THEN { do an adjustment }
19 318 1-  BEGIN
19 319 --          GetGlobalBounds(windRect);
19 320 --          newSize := windRect.botRight;
19 321 --          SubPt(windRect.topLeft, newSize);
19 322 --          AddPt(diff, newSize);
19 323 --          newSize := Point(PinRect(fResizeLimits, newSize));
19 324 --
19 325 --          SizeWindow(fWMgrWindow, newSize.h, newSize.v, FALSE);
19 326 -1  END;
19 327 -0 A  END { TWindow.AdaptToScreen };
19 328 --
19 329 --
19 330 --  {$$ MAREs}
19 331 --  {-----+
19 332 --  |  AllowsMenuAccess
19 333 --  +-----}
19 334 -- A  FUNCTION TWindow.AllowsMenuAccess: BOOLEAN;
19 335 --
19 336 0- A  BEGIN
19 337 --      { Default is to always allow menu access, even if modal }
19 338 --      AllowsMenuAccess := TRUE;
19 339 -0 A  END { TWindow.AllowsMenuAccess };
19 340 --
19 341 --
19 342 --  {$$ MAOpen}
19 343 --  {-----+
19 344 --  |  Center
19 345 --  +-----}
19 346 -- A  PROCEDURE TWindow.Center (horizontally, vertically: BOOLEAN);
19 347 --
19 348 --  VAR
19 349 --      screenSize:  Point;
19 350 --      globalbounds: Rect;
19 351 --
19 352 0- A  BEGIN
19 353 --      fHorzCentered := horizontally;
19 354 --      fVertCentered := vertically;
19 355 --      IF fWMgrWindow <> NIL THEN
19 356 1-  BEGIN
19 357 --          screenSize := screenBits.bounds.botRight;
19 358 --          SubPt(screenBits.bounds.topLeft, screenSize);
19 359 --          GetGlobalBounds(globalbounds);
19 360 --          WITH globalBounds DO
19 361 2-  BEGIN
19 362 --              IF horizontally THEN
19 363 --                  left := (screenSize.h - LengthRect(globalBounds, h)) DIV 2;
19 364 --              IF vertically THEN
19 365 --                  top := (screenSize.v - LengthRect(globalBounds, v)) DIV 2;
19 366 --                  MoveWindow(fWMgrWindow, left, top, FALSE);
19 367 -2  END;
19 368 -1  END;
19 369 -0 A  END { TWindow.Center };
19 370 --
19 371 --
19 372 --  {$$ MAClose}
19 373 --  {-----+
19 374 --  |  Close
19 375 --  +-----}
19 376 -- A  PROCEDURE TWindow.Close; OVERRIDE;
19 377 --
19 378 0- A  BEGIN
19 379 --      INHERITED Close;
19 380 --
19 381 --      IF gFrontWindow = SELF THEN
19 382 --          gFrontWindow := NIL;
19 383 --
19 384 --      Show(FALSE, kRedraw);
19 385 --      Activate(FALSE);           { ??? Is this necessary? }
19 386 --
19 387 --      IF fFreeOnClosing THEN
19 388 --          Free;
19 389 -0 A  END { TWindow.Close };
19 390 --
19 391 --
19 392 --  {$$ MAClose}
19 393 --  {-----+
19 394 --  |  CloseByUser
19 395 --  +-----}

```

```

19 396 -- A  PROCEDURE TWindow.CloseByUser;
19 397 --
19 398 --      VAR
19 399 --          openDocWindows:  INTEGER;
19 400 --
19 401 --      {-----+
19 402 --      |   CountOpenWindows   |
19 403 --      +-----}
19 404 -- B      PROCEDURE CountOpenWindows (aWindow: TWindow);
19 405 --
19 406 -- B      BEGIN
19 407 --          IF IsShown THEN
19 408 --              openDocWindows := openDocWindows + 1;
19 409 -- B      END;
19 410 --
19 411 -- A      BEGIN
19 412 --          IF fDocument = NIL THEN ( free window )
19 413 --              Close
19 414 --          ELSE
19 415 -- B              BEGIN
19 416 --                  { See how many open windows this document has }
19 417 --                  openDocWindows := 0;
19 418 --                  fDocument.ForAllWindowsDo(CountOpenWindows);
19 419 --
19 420 --                  IF fClosesDocument OR (openDocWindows = 1) THEN
19 421 --                      fDocument.Close
19 422 --                  ELSE
19 423 --                      Close;
19 424 -- B              END;
19 425 -- A      END ( TWindow.CloseByUser );
19 426 --
19 427 --
19 428 --      { $$ MRes }
19 429 --      {-----+
19 430 --      |   DoSetupMenus       |
19 431 --      +-----}
19 432 -- A      PROCEDURE TWindow.DoSetupMenus; OVERRIDE;
19 433 --
19 434 -- A      BEGIN
19 435 --          IF NOT fIsModal THEN          { Don't enable menu/app commands if modal }
19 436 --              INHERITED DoSetupMenus;
19 437 -- A      END ( TWindow.DoSetupMenus );
19 438 --
19 439 --
19 440 --      { $$ MRes }
19 441 --      {-----+
19 442 --      |   Draw               |
19 443 --      +-----}
19 444 -- A      PROCEDURE TWindow.Draw (area: Rect); OVERRIDE;
19 445 --
19 446 -- A      BEGIN
19 447 --          EraseRect(fWmgrWindow^.portRect);
19 448 -- A      END ( TWindow.Draw );
19 449 --
19 450 --
19 451 --      { $$ MRes }
19 452 --      {-----+
19 453 --      |   DrawContents       |
19 454 --      +-----}
19 455 -- A      PROCEDURE TWindow.DrawContents; OVERRIDE;
19 456 --
19 457 -- A      BEGIN
19 458 --          INHERITED DrawContents;
19 459 --          { Ensure that other views don't draw over the grow box by drawing it last }
19 460 --          IF fIsResizable AND Focus THEN
19 461 --              DrawResizeIcon;
19 462 -- A      END ( TWindow.DrawContents );
19 463 --
19 464 --
19 465 --      { $$ MRes }
19 466 --      {-----+
19 467 --      |   DrawResizeIcon     |
19 468 --      +-----}
19 469 -- A      PROCEDURE TWindow.DrawResizeIcon;
19 470 --
19 471 --      VAR
19 472 --          r:  Rect;
19 473 --
19 474 -- A      BEGIN
19 475 --          SetPort(fWmgrWindow);
19 476 --
19 477 --          { origin doesn't matter since we use the portRect }
19 478 --          r := thePort^.portRect;
19 479 --          r.left := r.right - kSBarSizeMinus1;
19 480 --          r.top := r.bottom - kSBarSizeMinus1;
19 481 --
19 482 --          { $IFC qDebug }
19 483 --          UseTempRgn('TWindow.DrawResizeIcon');
19 484 --          { $ENDC }
19 485 --
19 486 --          GetClip(gTempRgn);
19 487 --          ClipRect(r);
19 488 --          PenNormal;
19 489 --          DrawGrowIcon(fWmgrWindow);
19 490 --          SetClip(gTempRgn);
19 491 --
19 492 --          { $IFC qDebug }
19 493 --          DoneWithTempRgn;
19 494 --          { $ENDC }

```

```

19 495 -0 A   END { TWindow.DrawResizeIcon };
19 496 --
19 497 --
19 498 --      {$S MAREs}
19 499 --      {-----+
19 500 --      |   Focus
19 501 --      +-----}
19 502 -- A   FUNCTION TWindow.Focus: BOOLEAN; OVERRIDE;
19 503 --
19 504 --      VAR
19 505 --          r:          Rect;
19 506 --          aRgn:       RgnHandle;
19 507 --
19 508 0- A   BEGIN
19 509 --      IF qFocusedView = SELF THEN
19 510 1-          BEGIN
19 511 --          {$IFC qDebug}
19 512 --              IF LONGINT(fWmgrWindow^.portRect.topLeft) <> 0 THEN
19 513 --                  ProgramBreak('TWindow.Focus: Origin is not (0,0)');
19 514 --          {$ENDC}
19 515 --              Focus := IsShown;
19 516 1-          END
19 517 --      ELSE IF fWmgrWindow <> NIL THEN
19 518 1-          BEGIN
19 519 --              SetPort(fWmgrWindow);
19 520 --              SetOrigin(0, 0);
19 521 --              gLongOffset := gZeroVPt;
19 522 --              ClipRect(thePort^.portRect);
19 523 --              qFocusedView := SELF;
19 524 --              Focus := IsShown;
19 525 1-          END
19 526 --      ELSE
19 527 --          Focus := FALSE;
19 528 -0 A   END { TWindow.Focus };
19 529 --
19 530 --
19 531 --      {$S MAREs}
19 532 --      {-----+
19 533 --      |   FocusOnSuperView
19 534 --      +-----}
19 535 -- A   FUNCTION TWindow.FocusOnSuperView: BOOLEAN; OVERRIDE;
19 536 --
19 537 0- A   BEGIN
19 538 --      FocusOnSuperView := Focus;
19 539 -0 A   END { TWindow.FocusOnSuperView };
19 540 --
19 541 --
19 542 --      {$S MAOpen}
19 543 --      {-----+
19 544 --      |   ForceOnScreen
19 545 --      +-----}
19 546 -- A   PROCEDURE TWindow.ForceOnScreen;
19 547 --
19 548 --      CONST
19 549 --          here =          'ForceOnScreen';
19 550 --          GrayRgn =      $9EE;
19 551 --
19 552 --      VAR
19 553 --          global:         Rect;
19 554 --          tempRect:       Rect;
19 555 --          deltaH:         INTEGER;
19 556 --          deltaV:         INTEGER;
19 557 --          ptrToRgnHandle: ^RgnHandle;
19 558 --
19 559 0- A   BEGIN
19 560 --      fForcedOnScreen := TRUE;
19 561 --
19 562 --      GetGlobalBounds(global);
19 563 --
19 564 --      { Make sure the window is not too big. }
19 565 --      WITH global DO
19 566 1-          BEGIN
19 567 --              IF right - left > fResizeLimits.right THEN
19 568 --                  right := left + fResizeLimits.right - 1;
19 569 --
19 570 --              IF bottom - top > fResizeLimits.bottom THEN
19 571 --                  bottom := top + fResizeLimits.bottom - 1;
19 572 1-          END;
19 573 --
19 574 --          deltaH := 0;
19 575 --          deltaV := 0;
19 576 --
19 577 --          {$IFC qDebug}
19 578 --          UseTempRgn(here);
19 579 --          {$ENDC}
19 580 --
19 581 --          { On some systems (including the Radius FPD), it's possible to have a
19 582 --            non-rectangular desktop. Try to be nice to people who saved windows
19 583 --            on secondary screens. GrayRgn is the true indicator of the shape
19 584 --            of the desktop--screenBits.bounds is the size of the screen with the
19 585 --            menu bar on it. }
19 586 --          ptrToRgnHandle := POINTER(GrayRgn);
19 587 --          CopyRgn(ptrToRgnHandle^, gTempRgn);
19 588 --          InsetRgn(gTempRgn, gStdWScreenRect.left - 1,
19 589 --                  gStdWScreenRect.top - gMBarHeight - 1);
19 590 --
19 591 --          { RectInRgn sometimes lies on old ROMs, so if we're running on old ROMs
19 592 --            then always force the window into gStdWScreenBounds. }
19 593 --          IF NOT ((gConfiguration.hasROM128K) AND RectInRgn(global, gTempRgn)) THEN

```

```

19 594 --      WITH global DO
19 595 1-      BEGIN
19 596 --
19 597 --          IF bottom < gStdWScreenRect.top THEN
19 598 --              deltaV := gStdWScreenRect.top - top
19 599 --          ELSE IF top > gStdWScreenRect.bottom THEN
19 600 --              deltaV := gStdWScreenRect.bottom - top;
19 601 --
19 602 --          IF right < gStdWScreenRect.left THEN
19 603 --              deltaH := gStdWScreenRect.left - right
19 604 --          ELSE IF left > gStdWScreenRect.right THEN
19 605 --              deltaH := gStdWScreenRect.right - left;
19 606 --
19 607 -1      END;
19 608 --
19 609 --      ($IFC qDebug)
19 610 --      DoneWithTempRgn;
19 611 --      ($ENDC)
19 612 --
19 613 --      OffsetRect(global, deltaH, deltaV);
19 614 --      WITH global DO
19 615 1-      BEGIN
19 616 --          MoveWindow(fWMgrWindow, left, top, FALSE);
19 617 --          SizeWindow(fWMgrWindow, right - left, bottom - top, FALSE);
19 618 -1      END;
19 619 -0 A  END { TWindow.ForceOnScreen };
19 620 --
19 621 --
19 622 --      {$$ MARES}
19 623 --      {-----+
19 624 --      |   GetGlobalBounds   |
19 625 --      +-----+
19 626 -- A  PROCEDURE TWindow.GetGlobalBounds (VAR globalBounds: Rect);
19 627 --
19 628 --      VAR
19 629 --          savedPort:      GrafPtr;
19 630 --
19 631 0- A  BEGIN
19 632 --      IF fWMgrWindow <> NIL THEN
19 633 1-      BEGIN
19 634 --          GetPort(savedPort);
19 635 --          SetPort(fWMgrWindow);
19 636 --          globalBounds := fWMgrWindow^.portRect;
19 637 --          LocalToGlobal(globalBounds.topLeft);
19 638 --          LocalToGlobal(globalBounds.botRight);
19 639 --          SetPort(savedPort);
19 640 -1      END;
19 641 -0 A  END { TWindow.GetGlobalBounds };
19 642 --
19 643 --
19 644 --      {$IFC qInspector}
19 645 --      {$$ MAINspector}
19 646 --      {-----+
19 647 --      |   GetInspectorName   |
19 648 --      +-----+
19 649 -- A  PROCEDURE TWindow.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
19 650 --
19 651 0- A  BEGIN
19 652 --      IF (fWMgrWindow <> NIL) & (NOT ODD(ORD4(fWMgrWindow))) THEN
19 653 --          GetWTitle(fWMgrWindow, inspectorName);
19 654 -0 A  END { TWindow.GetInspectorName };
19 655 --      ($ENDC)
19 656 --
19 657 --
19 658 --      {$$ MARES}
19 659 --      {-----+
19 660 --      |   GetGrafPort   |
19 661 --      +-----+
19 662 -- A  FUNCTION TWindow.GetGrafPort: GrafPtr; OVERRIDE;
19 663 --
19 664 0- A  BEGIN
19 665 --      GetGrafPort := GrafPtr(fWMgrWindow);
19 666 -0 A  END { TWindow.GetGrafPort };
19 667 --
19 668 --
19 669 --      {$$ MARES}
19 670 --      {-----+
19 671 --      |   GetWindow   |
19 672 --      +-----+
19 673 -- A  FUNCTION TWindow.GetWindow: TWindow; OVERRIDE;
19 674 --
19 675 0- A  BEGIN
19 676 --      GetWindow := SELF;
19 677 -0 A  END { TWindow.GetWindow };
19 678 --
19 679 --
19 680 --      {$$ MASelCommand}
19 681 --      {-----+
19 682 --      |   HandleMouseDown   |
19 683 --      +-----+
19 684 -- A  FUNCTION TWindow.HandleMouseDown (theMouse:      VPoint;
19 685 --                                     VAR info:      EventInfo;
19 686 --                                     VAR hysteresis:  Point;
19 687 --                                     VAR theCommand:  TCommand): BOOLEAN; OVERRIDE;
19 688 --
19 689 --      VAR
19 690 --          wantsTheMouse:  BOOLEAN;
19 691 --          anEvent:        EventRecord;
19 692 --

```

```

19 693 0- A BEGIN
19 694 -- theCommand := gNoChanges;
19 695 --
19 696 -- wantsTheMouse := TRUE;
19 697 -- IF fWMgrWindow <> FrontWindow THEN
19 698 1- BEGIN
19 699 -- Select;
19 700 -- IF fDoFirstClick THEN
19 701 -- { Make sure this window is activated and updated }
19 702 -- WHILE gApplication.GetEvent(activMask, 0, NIL, anEvent) DO
19 703 -- gApplication.HandleEvent(anEvent)
19 704 -- ELSE
19 705 -- wantsTheMouse := FALSE;
19 706 -1 END;
19 707 --
19 708 -- IF wantsTheMouse THEN
19 709 -- HandleMouseDown := INHERITED HandleMouseDown(theMouse, info, hysteresis, theCommand)
19 710 -- ELSE
19 711 -- HandleMouseDown := FALSE;
19 712 -0 A END { TWindow.HandleMouseDown };
19 713 --
19 714 --
19 715 -- { $$ MAOpen }
19 716 -- {-----+
19 717 -- | InstallDocument |
19 718 -- +-----}
19 719 -- A PROCEDURE TWindow.InstallDocument (itsDocument: TDocument);
19 720 --
19 721 -- VAR
19 722 -- astring: Str255;
19 723 --
19 724 0- A BEGIN
19 725 -- fDocument := itsDocument;
19 726 -- IF itsDocument <> NIL THEN
19 727 1- BEGIN
19 728 -- gApplication.DeleteFreeWindow(SELF); { Just in case... }
19 729 -- itsDocument.AddWindow(SELF);
19 730 -- aString := itsDocument.fTitle; { SetTitleForDoc may move memory }
19 731 -- IF aString <> '' THEN { not an untitled document }
19 732 -- SetTitleForDoc(aString);
19 733 -- fNextHandler := itsDocument;
19 734 -1 END
19 735 -- ELSE
19 736 1- BEGIN
19 737 -- gApplication.AddFreeWindow(SELF);
19 738 -- fNextHandler := gApplication;
19 739 -1 END;
19 740 -0 A END { TWindow.InstallDocument };
19 741 --
19 742 --
19 743 -- { $$ MAREs }
19 744 -- {-----+
19 745 -- | IsShown |
19 746 -- +-----}
19 747 -- A FUNCTION TWindow.IsShown: BOOLEAN; OVERRIDE;
19 748 --
19 749 0- A BEGIN
19 750 -- IF fWMgrWindow <> NIL THEN
19 751 -- IsShown := WindowPeek(fWMgrWindow)^.visible
19 752 -- ELSE
19 753 -- IsShown := FALSE;
19 754 -0 A END { TWindow.IsShown };
19 755 --
19 756 --
19 757 -- { $$ MAREs }
19 758 -- {-----+
19 759 -- | Locate |
19 760 -- +-----}
19 761 -- A PROCEDURE TWindow.Locate (h, v: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
19 762 --
19 763 0- A BEGIN
19 764 -- IF fWMgrWindow <> NIL THEN
19 765 -- MoveWindow(fWMgrWindow, h, v, FALSE);
19 766 -0 A END { TWindow.Locate };
19 767 --
19 768 --
19 769 -- { $$ MAREs }
19 770 -- {-----+
19 771 -- | Select |
19 772 -- +-----}
19 773 -- A PROCEDURE TWindow.Select;
19 774 --
19 775 0- A BEGIN
19 776 -- gApplication.SelectWMgrWindow(fWMgrWindow);
19 777 -0 A END { TWindow.Select };
19 778 --
19 779 --
19 780 -- { $$ MAREs }
19 781 -- {-----+
19 782 -- | MoveByUser |
19 783 -- +-----}
19 784 -- A PROCEDURE TWindow.MoveByUser (globalMouse: Point);
19 785 --
19 786 -- VAR
19 787 -- boundsRect: Rect;
19 788 --
19 789 0- A BEGIN
19 790 -- boundsRect := fMoveBounds;
19 791 -- DragWindow(fWMgrWindow, globalMouse, boundsRect);

```

```

19 792 --      gLastUpTime := TickCount;    { needed for double clicking the title bar,
19 793 --                                     because DragWindow eats the mouseUpEvent }
19 794 -0 A  END { TWindow.MoveByUser };
19 795 --
19 796 --
19 797 --      {$S MAOpen}
19 798 --      {-----+
19 799 --      |   Open                               |
19 800 --      +-----+
19 801 -- A  PROCEDURE TWindow.Open; OVERRIDE;
19 802 --
19 803 0- A  BEGIN
19 804 --      WITH fWMgrWindow^.portRect DO
19 805 --          Resize(right - left, bottom - top, FALSE); { Be sure all views are sized right }
19 806 --          Show(TRUE, kRedraw);      { Make me visible }
19 807 --
19 808 --          INHERITED Open;             { Tell subviews to open in case they care }
19 809 -0 A  END { TWindow.Open };
19 810 --
19 811 --
19 812 --      {$S MANonRes}
19 813 --      {-----+
19 814 --      |   Resize                               |
19 815 --      +-----+
19 816 -- A  PROCEDURE TWindow.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
19 817 --
19 818 --      VAR
19 819 --          r:      Rect;
19 820 --
19 821 0- A  BEGIN
19 822 --          IF (width <> fSize.h) | (height <> fSize.v) THEN
19 823 1-      BEGIN
19 824 --          SizeWindow(fWMgrWindow, width, height, FALSE);
19 825 --
19 826 --          IF fIsResizable THEN { inval old and new positions of grow box }
19 827 --              IF Focus THEN
19 828 2-      BEGIN
19 829 --          SetRect(r, -kSBarSizeMinus1, -kSBarSizeMinus1, 0, 0);
19 830 --
19 831 --          OffsetRect(r, fSize.h, fSize.v);
19 832 --          EraseRect(r);
19 833 --          InvalidRect(r);
19 834 --
19 835 --          OffsetRect(r, width - fSize.h, height - fSize.v);
19 836 --          InvalidRect(r);
19 837 -2      END;
19 838 --
19 839 --          INHERITED Resize(width, height, invalidate);
19 840 -1      END;
19 841 -0 A  END { TWindow.Resize };
19 842 --
19 843 --
19 844 --      {$S MANonRes}
19 845 --      {-----+
19 846 --      |   ResizeByUser                               |
19 847 --      +-----+
19 848 -- A  PROCEDURE TWindow.ResizeByUser (globalMouse: Point);
19 849 --
19 850 --      VAR
19 851 --          growResult:    LONGINT;
19 852 --          resizeLimits:  Rect;
19 853 --
19 854 0- A  BEGIN
19 855 --          resizeLimits := fResizeLimits;      { GrowWindow may move SELF }
19 856 --          growResult := GrowWindow(fWMgrWindow, globalMouse, resizeLimits);
19 857 --          IF growResult <> 0 THEN
19 858 --              Resize(LoWrd(growResult), HiWrd(growResult), TRUE);
19 859 -0 A  END { TWindow.ResizeByUser };
19 860 --
19 861 --
19 862 --      {$S MAOpen}
19 863 --      {-----+
19 864 --      |   SetResizeLimits                               |
19 865 --      +-----+
19 866 -- A  PROCEDURE TWindow.SetResizeLimits (minSize, maxSize: Point);
19 867 --
19 868 --      TYPE
19 869 --          PWStateData = ^WStateData;
19 870 --          HWStateData = ^PWStateData;
19 871 --
19 872 0- A  BEGIN
19 873 --          { ??? Do we want to force the window's size to change if it is not
19 874 --              within the minimum and maximum size given above? }
19 875 --
19 876 --          fResizeLimits.topLeft := minSize;
19 877 --          fResizeLimits.botRight := maxSize;
19 878 --
19 879 --          IF BAND(fProcID, 8) <> 0 THEN
19 880 --              WITH HWStateData(WindowPeek(fWMgrWindow)^.DataHandle)^^.stdState DO
19 881 1-      BEGIN
19 882 --              right := Min(right, left + maxSize.h);
19 883 --              bottom := Min(bottom, top + maxSize.v);
19 884 -1      END;
19 885 -0 A  END { TWindow.SetResizeLimits };
19 886 --
19 887 --
19 888 --      {$S MRes}
19 889 --      {-----+
19 890 --      |   SetTarget

```

```

19 891 -- +-----}
19 892 -- A  PROCEDURE TWindow.SetTarget (newTarget: TEvtHandler);
19 893 --
19 894 0- A  BEGIN
19 895 --         fTarget := newTarget;
19 896 --         IF gFrontWindow = SELF THEN
19 897 --             gApplication.SetTarget(newTarget);
19 898 -- A  END ( TWindow.SetTarget );
19 899 --
19 900 --
19 901 --         {$S MAFile}
19 902 --         {-----+
19 903 --         |   SetTitleForDoc
19 904 --         +-----}
19 905 -- A  PROCEDURE TWindow.SetTitleForDoc (newDocTitle: Str255);
19 906 --
19 907 --         VAR
19 908 --             title:          Str255;
19 909 --
19 910 0- A  BEGIN
19 911 --         IF fPreDocname > 0 THEN { optimize for case of nothing to do }
19 912 1-   BEGIN
19 913 --             GetWTitle(fWmgrWindow, title);
19 914 --
19 915 --             IF SubstituteInTitle(title, newDocTitle, fPreDocname, fConstTitle) THEN
19 916 --                 SetWTitle(fWmgrWindow, title);
19 917 1-   END;
19 918 -- A  END ( TWindow.SetTitleForDoc );
19 919 --
19 920 --
19 921 --         {$S MANonRes}
19 922 --         {-----+
19 923 --         |   Show
19 924 --         +-----}
19 925 -- A  PROCEDURE TWindow.Show (state, redraw: BOOLEAN); OVERRIDE;
19 926 --
19 927 0- A  BEGIN
19 928 --         IF (fWmgrWindow <> NIL) AND redraw THEN
19 929 --             IF state THEN
19 930 --                 ShowWindow(fWmgrWindow)
19 931 --             ELSE
19 932 --                 HideWindow(fWmgrWindow);
19 933 --             fShown := state;
19 934 -- A  END;
19 935 --
19 936 --
19 937 --         {$S MAOpen}
19 938 --         {-----+
19 939 --         |   SimpleStagger
19 940 --         +-----}
19 941 -- A  PROCEDURE TWindow.SimpleStagger (dh, dv: INTEGER; VAR counter: INTEGER);
19 942 --
19 943 --         VAR
19 944 --             posRect:      Rect;      { The topLeft of the window must fall in here. }
19 945 --             pt:           Point;
19 946 --             nSlots:       INTEGER;
19 947 --             slot:         INTEGER;
19 948 --             wTopLeft:     Point;
19 949 --
19 950 0- A  BEGIN
19 951 --             fStaggered := TRUE;
19 952 --             { Get window rect, which gives us posRect.topLeft & size of window }
19 953 --             GetGlobalBounds(posRect);
19 954 --             wTopLeft := posRect.topLeft;
19 955 --
19 956 --             { Compute posRect.botRight = wTopLeft + <bounds botRight> - <window botRight> }
19 957 --             pt := wTopLeft;
19 958 --             AddPt(fMoveBounds.botRight, pt);
19 959 --             SubPt(posRect.botRight, pt);
19 960 --             posRect.botRight := pt;
19 961 --
19 962 --             { This covers the case of dh & dv both >= 0; if either is less than
19 963 --             0, the right (bottom) limit is set to the left (top) edge of the
19 964 --             screen -- allowing for some margin. This makes the right (bottom)
19 965 --             edge less than the left (top) edge, but this is OK since the DIV
19 966 --             statement below will be dividing 2 negative numbers. }
19 967 --             IF dh < 0 THEN
19 968 --                 posRect.right := fMoveBounds.left;
19 969 --             IF dv < 0 THEN
19 970 --                 posRect.bottom := fMoveBounds.top;
19 971 --
19 972 --             { This code avoids divide by zero problems }
19 973 --             IF (dh = 0) OR (dv = 0) THEN
19 974 --                 nSlots := 0
19 975 --             ELSE
19 976 --                 nSlots := Min((posRect.right-posRect.left+dh-1) DIV dh,
19 977 --                               (posRect.bottom-posRect.top+dv-1) DIV dv );
19 978 --             IF nSlots = 0 THEN
19 979 --                 slot := 0
19 980 --             ELSE
19 981 --                 slot := counter MOD nSlots;
19 982 --
19 983 --             IF slot <> 0 THEN { move the window }
19 984 1-   BEGIN
19 985 --                 { pt ends up as the place to position the window }
19 986 --                 pt := wTopLeft;
19 987 --
19 988 --                 pt.h := pt.h + (slot * dh);
19 989 --                 pt.v := pt.v + (slot * dv);

```

```

19 990 --
19 991 --         MoveWindow(fWmgrWindow, pt.h, pt.v, FALSE);
19 992 --1         END;
19 993 --
19 994 --         counter := counter + 1;
19 995 --0 A     END { TWindow.SimpleStagger };
19 996 --
19 997 --
19 998 --         {$S MANonRes}
19 999 --         {-----+
19 1000 --         |   Zoom
19 1001 --         +-----}
19 1002 -- A     PROCEDURE TWindow.Zoom (partCode: INTEGER);
19 1003 --
19 1004 --0 A     BEGIN
19 1005 --         { Erasing was suggested by Ernie B.; the Finder does this.
19 1006 --         { The ROM has a bug requiring that thePort be the window
19 1007 --         { being zoomed. }
19 1008 --         IF Focus THEN { ??? This may not be necessary }
19 1009 --1         BEGIN
19 1010 --             EraseRect(thePort^.portRect);
19 1011 --             ZoomWindow(fWmgrWindow, partCode, FALSE);
19 1012 --
19 1013 --             WITH fWmgrWindow^.portRect DO
19 1014 --                 Resize(right - left, bottom - top, TRUE);
19 1015 --1         END;
19 1016 --0 A     END { TWindow.Zoom };
19 1017 --
19 1018 --         {$IFC qDebug}
19 1019 --         {$S MAFields}
19 1020 --         {-----+
19 1021 --         |   Fields
19 1022 --         +-----}
19 1023 -- A     PROCEDURE TWindow.Fields (PROCEDURE DoToField (fieldName: Str255;
19 1024 --                                     fieldAddr: Ptr;
19 1025 --                                     fieldType: INTEGER)); OVERRIDE;
19 1026 --
19 1027 --         VAR
19 1028 --             theFlag:      BOOLEAN;
19 1029 --0 A     BEGIN
19 1030 --         DoToField('TWindow', NIL, bClass);
19 1031 --         DoToField('fWmgrWindow', @fWmgrWindow, bWindowPtr);
19 1032 --         DoToField('fProcID', @fProcID, bInteger);
19 1033 --         DoToField('fMoveBounds', @fMoveBounds, bRect);
19 1034 --         DoToField('fResizeLimits', @fResizeLimits, bRect);
19 1035 --         DoToField('fTarget', @fTarget, bObject);
19 1036 --         DoToField('fTargetID', @fTargetID, bOSType);
19 1037 --         DoToField('fPreDocName', @fPreDocName, bInteger);
19 1038 --         DoToField('fConstTitle', @fConstTitle, bInteger);
19 1039 --         DoToField('fAdapted', @fAdapted, bBoolean);
19 1040 --         DoToField('fHorzCentered', @fHorzCentered, bBoolean);
19 1041 --         DoToField('fVertCentered', @fVertCentered, bBoolean);
19 1042 --         DoToField('fStaggered', @fStaggered, bBoolean);
19 1043 --         DoToField('fForcedOnScreen', @fForcedOnScreen, bBoolean);
19 1044 --         DoToField('fisActive', @fisActive, bBoolean);
19 1045 --         DoToField('fisResizable', @fisResizable, bBoolean);
19 1046 --         DoToField('fisClosable', @fisClosable, bBoolean);
19 1047 --         DoToField('fFreeOnClosing', @fFreeOnClosing, bBoolean);
19 1048 --         DoToField('fDisposeOnFree', @fDisposeOnFree, bBoolean);
19 1049 --         DoToField('fClosesDocument', @fClosesDocument, bBoolean);
19 1050 --         DoToField('fOpenInitially', @fOpenInitially, bBoolean);
19 1051 --         DoToField('fisModal', @fisModal, bBoolean);
19 1052 --         DoToField('fDoFirstClick', @fDoFirstClick, bBoolean);
19 1053 --         INHERITED Fields(DoToField);
19 1054 --0 A     END { TWindow.Fields };
19 1055 --         {$ENDC}
13 3007 --         {$I UMacApp.TScroller.p}

```



```

20 1 -- {SP}
20 2 -- { UMacApp.TScroller.p }
20 3 -- { Copyright © 1987, 1988 by Apple Computer Inc. All rights reserved. }
20 4 --
20 5 -- (*****
20 6 -- (* T S c r o l l e r *)
20 7 -- (*****
20 8 -- {-----+
20 9 -- | IScroller |
20 10 -- +-----}
20 11 -- {IFC qProceduralViews}
20 12 -- {$S MAOpen}
20 13 -- A PROCEDURE TScroller.IScroller (itsSuperView: TView; itsLocation, itsSize: VPoint;
20 14 -- itsHSizeDet, itsVSizeDet: SizeDeterminer;
20 15 -- itsHorzMax, itsVertMax: VCoordinate;
20 16 -- wantHorzSBar, wantVertSBar: BOOLEAN);
20 17 -- VAR
20 18 -- scrollLimit: VPoint;
20 19 --
20 20 -- A BEGIN
20 21 -- fScrollBars[h] := NIL;
20 22 -- fScrollBars[v] := NIL;
20 23 --
20 24 -- IView(NIL, itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
20 25 --
20 26 -- fTranslation := gZeroVpt;
20 27 -- fMaxTranslation := gZeroVpt;
20 28 -- fSBarOffsets := gZeroVRect;
20 29 --
20 30 -- SetScrollParameters(kStdScrollUnit, kStdScrollUnit, FALSE, FALSE);
20 31 --
20 32 -- scrollLimit.h := itsHorzMax;
20 33 -- scrollLimit.v := itsVertMax;
20 34 -- SetScrollLimits(scrollLimit, kDontRedraw);
20 35 --
20 36 -- CreateScrollBars(wantHorzSBar, wantVertSBar);
20 37 --
20 38 -- A END {TScroller.IScroller};
20 39 -- {$ENDC}
20 40 --
20 41 -- {-----+
20 42 -- | IRes |
20 43 -- +-----}
20 44 -- {IFC qTemplateViews}
20 45 -- {$S MAOpen}
20 46 -- A PROCEDURE TScroller.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr);
20 47 --
20 48 -- VAR
20 49 -- scrollLimit: VPoint;
20 50 --
20 51 -- A BEGIN
20 52 -- fScrollBars[h] := NIL;
20 53 -- fScrollBars[v] := NIL;
20 54 --
20 55 -- INHERITED IRes(NIL, itsSuperView, itsParams);
20 56 --
20 57 -- WITH ScrollerTemplatePtr(itsParams)^ DO
20 58 -- BEGIN
20 59 -- fTranslation := gZeroVpt;
20 60 -- scrollLimit.h := horzMax;
20 61 -- scrollLimit.v := vertMax;
20 62 -- SetScrollLimits(scrollLimit, kDontRedraw);
20 63 --
20 64 -- SetScrollParameters(hScrollUnits, vScrollUnits, hConstrain, vConstrain);
20 65 -- {$H-}
20 66 -- RectToVRect(sBarOffsets, fSBarOffsets);
20 67 -- {$H+}
20 68 -- CreateScrollBars(wantHSBar, wantVSBar);
20 69 -- END;
20 70 --
20 71 -- OffsetPtr(itsParams, SIZEOF(ScrollerTemplate));
20 72 -- A END;
20 73 -- {$ENDC}
20 74 --
20 75 -- {IFC qWriteTemplates}
20 76 -- {$S MAWriteRes}
20 77 -- {-----+
20 78 -- | WRes |
20 79 -- +-----}
20 80 -- A PROCEDURE TScroller.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
20 81 --
20 82 -- VAR
20 83 -- scPtr: ScrollerTemplatePtr;
20 84 --
20 85 -- A BEGIN
20 86 -- INHERITED WRes(theResource, itsParams);
20 87 --
20 88 -- scPtr := ScrollerTemplatePtr(ExpandPtr(theResource, itsParams,
20 89 -- SIZEOF(ScrollerTemplate)));
20 90 --
20 91 -- WITH scPtr^ DO
20 92 -- BEGIN
20 93 -- wantVSBar := fScrollBars[h] <> NIL;
20 94 -- wantHSBar := fScrollBars[v] <> NIL;
20 95 -- vertMax := fScrollLimit.v;
20 96 -- horzMax := fScrollLimit.h;
20 97 -- vScrollUnits := fScrollUnit.v;
20 98 -- hScrollUnits := fScrollUnit.h;

```

```

20 99 --          vConstrain := fConstrain[v];
20 100 --          hConstrain := fConstrain[h];
20 101 --          VRectToRect(fSBarOffsets, sBarOffsets);
20 102 --          END;
20 103 -- A      END;
20 104 --          {$ENDC}
20 105 --
20 106 --          {$IFC qWriteTemplates}
20 107 --          {$S MAWriteRes}
20 108 --          {-----+
20 109 --          |   WriteRes
20 110 --          +-----}
20 111 -- A      PROCEDURE TScroller.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr): OVERRIDE;
20 112 -- A      BEGIN
20 113 --          gWResSignature := 'scrl';   gWResType := 'TScroller';
20 114 --          WRes(theResource, itsParams);
20 115 -- A      END;
20 116 --          {$ENDC}
20 117 --
20 118 --          {-----+
20 119 --          |   Free
20 120 --          +-----}
20 121 --          {$S MAClose}
20 122 -- A      PROCEDURE TScroller.Free; OVERRIDE;
20 123 --
20 124 -- A      BEGIN
20 125 --          IF fScrollBars[h] <> NIL THEN
20 126 --              fScrollBars[h].Free;
20 127 --          IF fScrollBars[v] <> NIL THEN
20 128 --              fScrollBars[v].Free;
20 129 --
20 130 --          INHERITED Free;
20 131 -- A      END (TScroller.Free);
20 132 --
20 133 --          {-----+
20 134 --          |   AddSubView
20 135 --          +-----}
20 136 --          {$S MAOpen}
20 137 -- A      PROCEDURE TScroller.AddSubView (theSubView: TView); OVERRIDE;
20 138 --
20 139 -- A      BEGIN
20 140 --          INHERITED AddSubView(theSubView);
20 141 --          theSubView.BeInScroller(SELF);
20 142 -- A      END (TScroller.AddSubView);
20 143 --
20 144 --          {-----+
20 145 --          |   AdjustScrollBars
20 146 --          +-----}
20 147 --          {$S MANonRes}
20 148 -- A      PROCEDURE TScroller.AdjustScrollBars;
20 149 --
20 150 --          VAR
20 151 --              vhs:          VHSelect;
20 152 --              aScrollBar:   TScrollBar;
20 153 --              ortho:        VHSelect;
20 154 --              loc:          VPoint;
20 155 --              size:         VPoint;
20 156 --
20 157 -- A      BEGIN
20 158 --          FOR vhs := v TO h DO
20 159 --              BEGIN
20 160 --                  aScrollBar := fScrollBars[vhs];
20 161 --                  IF aScrollBar <> NIL THEN
20 162 --                      BEGIN
20 163 --                          ortho := gOrthogonal[vhs];
20 164 --                          loc := fLocation;
20 165 --                          size := fSize;
20 166 --                          loc.vh[vhs] := loc.vh[vhs] + fSBarOffsets.topLeft.vh[vhs] - 1;
20 167 --                          loc.vh[gOrthogonal[vhs]] := loc.vh[ortho] + size.vh[ortho];
20 168 --                          size.vh[vhs] := size.vh[vhs] - fSBarOffsets.topLeft.vh[vhs] +
20 169 --                              fSBarOffsets.botRight.vh[vhs] + 2;
20 170 --                          size.vh[ortho] := kSBarSize;
20 171 --
20 172 --                          aScrollBar.Resize(size.h, size.v, TRUE);
20 173 --                          aScrollBar.Locate(loc.h, loc.v, TRUE);
20 174 --                      END;
20 175 --              END;
20 176 -- A      END (TScroller.AdjustScrollBars);
20 177 --
20 178 --          {-----+
20 179 --          |   AutoScroll
20 180 --          +-----}
20 181 --          {$S MADoCommand}
20 182 -- A      PROCEDURE TScroller.AutoScroll (viewPt: VPoint; VAR delta: VPoint);
20 183 --
20 184 --          VAR
20 185 --              vhs:          VHSelect;
20 186 --              myExtent:     VRect;
20 187 --
20 188 -- A      BEGIN
20 189 --          {??? Do we want to autoscroll if the edges of the frame are covered?}
20 190 --          {??? What about scrolling when the grow icon is in front?}
20 191 --
20 192 --          delta := gZeroVPt;
20 193 --          GetExtent(myExtent);
20 194 --          FOR vhs := v TO h DO
20 195 --              IF viewPt.vh[vhs] < myExtent.topLeft.vh[vhs] THEN
20 196 --                  delta.vh[vhs] := Max(-fScrollUnit.vh[vhs], -fTranslation.vh[vhs])
20 197 --              ELSE IF viewPt.vh[vhs] > myExtent.botRight.vh[vhs] THEN

```

```

20 198 --          delta.vh[vhs] := Min(fScrollUnit.vh[vhs],
20 199 --              fMaxTranslation.vh[vhs] - fTranslation.vh[vhs]);
20 200 -0 A  END (TScroller.AutoScroll);
20 201 --
20 202 --  {-----+
20 203 --  | CreateScrollBars
20 204 --  +-----}
20 205 --  {$S MAOpen}
20 206 -- A  PROCEDURE TScroller.CreateScrollBars (wantHorzSBar, wantVertSBar: BOOLEAN);
20 207 --
20 208 --  VAR
20 209 --      sBarLocation,
20 210 --      sBarSize:      VPoint;
20 211 --      anSScrollBar:  TSScrollBar;
20 212 --      fi:            FailInfo;
20 213 --
20 214 -- B  PROCEDURE HandleFailure (error: OSErr; message: LONGINT);
20 215 0- B  BEGIN
20 216 --      Free;
20 217 -0 B  END;
20 218 --
20 219 0- A  BEGIN
20 220 --      CatchFailures(fi, HandleFailure);
20 221 --
20 222 --      {??? Perhaps use AdjustScrollBars?, or local nested routine to create sbar?}
20 223 --      IF wantHorzSBar THEN
20 224 1-  BEGIN
20 225 --          SetVpt(sBarLocation, fLocation.h + fSBarOffsets.left - 1, fLocation.v + fSize.v);
20 226 --          SetVpt(sBarSize, fSize.h - fSBarOffsets.left + fSBarOffsets.right + 2, kSBarSize);
20 227 --          New(anSScrollBar);
20 228 --          FailNIL(anSScrollBar);
20 229 --          anSScrollBar.ISScrollBar(fSuperView, sBarLocation, sBarSize, SizeVariable,
20 230 --              SizeVariable, h, fMaxTranslation.h, SELF);
20 231 -1  END
20 232 --      ELSE
20 233 --          fScrollBars[h] := NIL;
20 234 --
20 235 --      IF wantVertSBar THEN
20 236 1-  BEGIN
20 237 --          SetVpt(sBarLocation, fLocation.h + fSize.h, fLocation.v + fSBarOffsets.top - 1);
20 238 --          SetVpt(sBarSize, kSBarSize, fSize.v - fSBarOffsets.top + fSBarOffsets.bottom + 2);
20 239 --          New(anSScrollBar);
20 240 --          FailNIL(anSScrollBar);
20 241 --          anSScrollBar.ISScrollBar(fSuperView, sBarLocation, sBarSize, SizeVariable,
20 242 --              SizeVariable, v, fMaxTranslation.v, SELF);
20 243 -1  END
20 244 --      ELSE
20 245 --          fScrollBars[v] := NIL;
20 246 --
20 247 --      Success(fi);
20 248 -0 A  END (TScroller.CreateScrollBars);
20 249 --
20 250 --  {-----+
20 251 --  | DoScroll
20 252 --  +-----}
20 253 --  {$S MAScroll}
20 254 -- A  PROCEDURE TScroller.DoScroll (delta: VPoint; redraw: BOOLEAN);
20 255 --
20 256 --  VAR
20 257 --      vhs:          VHSelect;
20 258 --      theWindow:    TWindow;
20 259 --
20 260 0- A  BEGIN
20 261 --      FOR vhs := v TO h DO
20 262 --          IF delta.vh[vhs] < 0 THEN
20 263 --              delta.vh[vhs] := Max(delta.vh[vhs], -fTranslation.vh[vhs])
20 264 --          ELSE IF delta.vh[vhs] > 0 THEN
20 265 --              delta.vh[vhs] := Min(delta.vh[vhs],
20 266 --                  fMaxTranslation.vh[vhs] - fTranslation.vh[vhs]);
20 267 --
20 268 --          IF (delta.h <> 0) | (delta.v <> 0) THEN
20 269 1-  BEGIN
20 270 --              IF redraw THEN
20 271 2-  BEGIN
20 272 --                  theWindow := GetWindow;
20 273 --                  IF NOT EmptyRgn(WindowPeek(theWindow.fWMgrWindow)^.updateRgn) THEN
20 274 --                      theWindow.Update;
20 275 -2  END;
20 276 --
20 277 --              {$H-}
20 278 --                  AddVpt(delta, fTranslation);
20 279 --              {$H+}
20 280 --              IF gFocusedView = SELF THEN
20 281 --                  gLongOffset := fTranslation;
20 282 --              IF redraw THEN
20 283 --                  ScrollDraw(delta);
20 284 --              END;
20 285 -1  END;
20 286 -0 A  END (TScroller.DoScroll);
20 287 --
20 288 --
20 289 --  {$IFC qDebug}
20 290 --  {-----+
20 291 --  | Fields
20 292 --  +-----}
20 293 --  {$S MAFields}
20 294 -- A  PROCEDURE TScroller.Fields (PROCEDURE DoToField (fieldName: Str255;
20 295 --      fieldAddr: Ptr;
20 296 --      fieldType: INTEGER)); OVERRIDE;

```

```

20 297 --
20 298 0- A      BEGIN
20 299 --          DoToField('TScroller', NIL, bClass);
20 300 --          DoToField('fTranslation', @fTranslation, bVPoint);
20 301 --          DoToField('fScrollLimit', @fScrollLimit, bVPoint);
20 302 --          DoToField('fMaxTranslation', @fMaxTranslation, bVPoint);
20 303 --          DoToField('fScrollBars[v]', @fScrollBars[v], bObject);
20 304 --          DoToField('fScrollBars[h]', @fScrollBars[h], bObject);
20 305 --          DoToField('fScrollUnit', @fScrollUnit, bPoint);
20 306 --          DoToField('fSBarOffsets', @fSBarOffsets, bVRect);
20 307 --          DoToField('fConstrain[h]', @fConstrain[h], bBoolean);
20 308 --          DoToField('fConstrain[v]', @fConstrain[v], bBoolean);
20 309 --          INHERITED Fields(DoToField);
20 310 -0 A      END {TScroller.Fields};
20 311 --      {$ENDC}
20 312 --
20 313 --      {-----+
20 314 --      | Focus
20 315 --      +-----+
20 316 --      {$S Mares}
20 317 -- A      FUNCTION TScroller.Focus: BOOLEAN;
20 318 --
20 319 0- A      BEGIN
20 320 --          Focus := INHERITED Focus;
20 321 --          gLongOffset := fTranslation
20 322 -0 A      END {TScroller.Focus};
20 323 --
20 324 --      {-----+
20 325 --      | GetExtent
20 326 --      +-----+
20 327 --      {$S Mares}
20 328 -- A      PROCEDURE TScroller.GetExtent (VAR itsExtent: VRect); OVERRIDE;
20 329 --
20 330 0- A      BEGIN
20 331 --          INHERITED GetExtent(itsExtent);
20 332 --          OffsetVRect(itsExtent, fTranslation.h, fTranslation.v);
20 333 -0 A      END {TScroller.GetExtent};
20 334 --
20 335 --      {-----+
20 336 --      | GetScroller
20 337 --      +-----+
20 338 --      {$S Mares}
20 339 -- A      FUNCTION TScroller.GetScroller (immediateSuperview: BOOLEAN): TScroller;
20 340 --
20 341 0- A      BEGIN
20 342 --          GetScroller := SELF;
20 343 -0 A      END {TScroller.GetScroller};
20 344 --
20 345 --      {-----+
20 346 --      | HaveScrollBar
20 347 --      +-----+
20 348 --      {$S Mares}
20 349 -- A      PROCEDURE TScroller.HaveScrollBar (theScrollBar: TSScrollBar;
20 350 --          direction: VHSelect);
20 351 --
20 352 0- A      BEGIN
20 353 --          {$IFC qDebug}
20 354 --              IF (theScrollBar <> NIL) & (theScrollBar.fDirection <> direction) THEN
20 355 --                  ProgramBreak('TScroller.HaveScrollBar: Scroll bar is wrong direction.');
```

```

20 396 --      ($$ MANonRes)
20 397 -- A  PROCEDURE TScroller.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
20 398 --
20 399 --      VAR
20 400 --          vhs:          VHSelect;
20 401 --          sBarWasVisible:  ARRAY [VHSelect] OF BOOLEAN;
20 402 --
20 403 0- A  BEGIN
20 404 --      { If the scroll bars are visible, erase them now so that they aren't temporarily
20 405 --        displayed in the wrong place.}
20 406 --      FOR vhs := v TO h DO
20 407 --          IF (fScrollBars[vhs] <> NIL) & fScrollBars[vhs].Focus & (fScrollBars[vhs].fCmqrControl^.contrlVis <> 0) THEN
20 408 1-              BEGIN
20 409 --                  sBarWasVisible[vhs] := TRUE;
20 410 --                  HideControl(fScrollBars[vhs].fCmqrControl);
20 411 -1              END
20 412 --          ELSE
20 413 --              sBarWasVisible[vhs] := FALSE;
20 414 --
20 415 --      INHERITED Resize(width, height, invalidate);
20 416 --      AdjustScrollBars;
20 417 --      SetScrollLimits(fScrollLimit, kDontRedraw); (Readjust sbar maximums and fMaxTranslation)
20 418 --
20 419 --      FOR vhs := v TO h DO
20 420 --          IF sBarWasVisible[vhs] THEN
20 421 --              fScrollBars[vhs].fCmqrControl^.contrlVis := 1;
20 422 -0 A  END (TScroller.Resize);
20 423 --
20 424 --      {-----+
20 425 --      |   RevealRect   |
20 426 --      +-----+}
20 427 --      {$$ MAScroll}
20 428 -- A  PROCEDURE TScroller.RevealRect (rectToReveal: VRect; minToSee: Point;
20 429 --                                   redraw: BOOLEAN); OVERRIDE;
20 430 --
20 431 --      VAR
20 432 --          myExtent:      VRect;
20 433 --          vhs:           VHSelect;
20 434 --          minAmt:        VCoordinate;
20 435 --          coord:         VCoordinate;
20 436 --          delta:         VPoint;
20 437 --
20 438 0- A  BEGIN
20 439 --          GetExtent(myExtent);
20 440 --
20 441 --          FOR vhs := v TO h DO
20 442 1-              BEGIN
20 443 --                  minAmt := MIN(LengthVRect(myExtent, vhs), minToSee.vh[vhs]);
20 444 --                  coord := rectToReveal.topLeft.vh[vhs] + minAmt - myExtent.botRight.vh[vhs];
20 445 --                  IF coord <= 0 THEN
20 446 --                      coord := Min(0, rectToReveal.botRight.vh[vhs] - minAmt - myExtent.topLeft.vh[vhs]);
20 447 --                      delta.vh[vhs] := coord;
20 448 -1                  END;
20 449 --
20 450 --                  ScrollBy(delta.h, delta.v, redraw);
20 451 -0 A  END (TScroller.RevealRect);
20 452 --
20 453 --      {-----+
20 454 --      |   ScrollBy   |
20 455 --      +-----+}
20 456 --      {$$ MAScroll}
20 457 -- A  PROCEDURE TScroller.ScrollBy (deltaH, deltaV: VCoordinate; redraw: BOOLEAN);
20 458 --
20 459 --      VAR
20 460 --          delta:         VPoint;
20 461 --
20 462 0- A  BEGIN
20 463 --          IF (deltaH <> 0) | (deltaV <> 0) THEN
20 464 1-              BEGIN
20 465 --                  IF fScrollBars[v] <> NIL THEN
20 466 --                      fScrollBars[v].DeltaValue(deltaV);
20 467 --                  IF fScrollBars[h] <> NIL THEN
20 468 --                      fScrollBars[h].DeltaValue(deltaH);
20 469 --
20 470 --                  SetVpt(delta, deltaH, deltaV);
20 471 --                  DoScroll(delta, redraw);
20 472 -1              END;
20 473 -0 A  END (TScroller.ScrollBy);
20 474 --
20 475 --      {-----+
20 476 --      |   ScrollDraw   |
20 477 --      +-----+}
20 478 --      {$$ MAScroll}
20 479 -- A  PROCEDURE TScroller.ScrollDraw (delta: VPoint);
20 480 --
20 481 --      VAR
20 482 --          myExtent:      VRect;
20 483 --          r:             Rect;
20 484 --
20 485 0- A  BEGIN
20 486 --          IF Focus THEN
20 487 1-              BEGIN
20 488 --                  GetExtent(myExtent);
20 489 --                  ViewToQDRect(myExtent, r); (!!! GetQDExtent doesn't work for scrollers)
20 490 --
20 491 --                  ($IFC qDebug)
20 492 --                      IF gIntenseDebugging THEN
20 493 --                          BEGIN
20 494 2-

```

```

20 495 --      WritelnRect('          r', r);
20 496 --      Writeln;
20 497 --      WritelnVpt('          gLongOffset', gLongOffset);
20 498 --      Writeln;
20 499 --      END;
20 500 --  {$ENDC}
20 501 --
20 502 --      IF (ABS(delta.h) > kMaxCoord) | (ABS(delta.v) > kMaxCoord) THEN
20 503 --          InvalidRect(r)
20 504 --      ELSE
20 505 --          BEGIN
20 506 --              {$IFC qDebug}
20 507 --                  UseTempRgn('TScroller.ScrollDraw');
20 508 --              {$ENDC}
20 509 --                  ScrollRect(r, -delta.h, -delta.v, gTempRgn);
20 510 --                  InvalRgn(gTempRgn);
20 511 --              {$IFC qDebug}
20 512 --                  DoneWithTempRgn;
20 513 --              {$ENDC}
20 514 --          END;
20 515 --      END;
20 516 --
20 517 --      BeginUpdate(thePort);  (Use thePort because we know we're focused)
20 518 --      EraseRect(thePort^.portRect);
20 519 --      DrawContents;
20 520 --      EndUpdate(thePort);
20 521 --      END;
20 522 -- 0 A  END {TScroller.ScrollDraw};
20 523 --
20 524 --      {-----+
20 525 --      |  ScrollRelative  |
20 526 --      +-----}
20 527 --      {$S MAScroll}
20 528 -- 0 A  FUNCTION TScroller.ScrollRelative (vhs: VHSelect; sBarValue: VCoordinate): VCoordinate;
20 529 --
20 530 --      VAR
20 531 --          pixelDelta:    VPoint;
20 532 --          newValue:      VCoordinate;
20 533 --
20 534 -- 0- A  BEGIN
20 535 --      IF fConstrain[vhs] & (sBarValue <> fMaxTranslation.vh[vhs]) THEN
20 536 --          newValue := (sBarValue + fScrollUnit.vh[vhs] DIV 2) DIV fScrollUnit.vh[vhs] *
20 537 --                      fScrollUnit.vh[vhs]
20 538 --      ELSE
20 539 --          newValue := sBarValue;
20 540 --
20 541 --          pixelDelta := gZeroVpt;
20 542 --          pixelDelta.vh[vhs] := newValue - fTranslation.vh[vhs];
20 543 --          DoScroll(pixelDelta, kRedraw);
20 544 --          ScrollRelative := newValue - sBarValue;
20 545 -- 0 A  END {TScroller.ScrollRelative};
20 546 --
20 547 --      {-----+
20 548 --      |  ScrollStep  |
20 549 --      +-----}
20 550 --      {$S MAScroll}
20 551 -- 0 A  FUNCTION TScroller.ScrollStep (vhs: VHSelect; partCode: INTEGER): VCoordinate;
20 552 --
20 553 --      VAR
20 554 --          scrollUnit:    INTEGER;
20 555 --          delta:        VPoint;
20 556 --          frameSize:    VCoordinate;
20 557 --
20 558 -- 0- A  BEGIN
20 559 --          scrollUnit := fScrollUnit.vh[vhs];
20 560 --          delta := gZeroVpt;
20 561 --          CASE partCode OF
20 562 --              inUpButton,
20 563 --              inDownButton:
20 564 --                  delta.vh[vhs] := scrollUnit;
20 565 --              inPageUp,
20 566 --              inPageDown:
20 567 --                  BEGIN
20 568 --                      frameSize := fSize.vh[vhs];
20 569 --                      IF fConstrain[vhs] THEN
20 570 --                          frameSize := frameSize - (frameSize MOD scrollUnit);
20 571 --                      delta.vh[vhs] := frameSize;
20 572 --                  END;
20 573 --          {$IFC qDebug}
20 574 --              OTHERWISE
20 575 --                  BEGIN
20 576 --                      Writeln('TScroller.ScrollStep: bad part code=', partCode:0);
20 577 --                      ProgramBreak('');
20 578 --                  END;
20 579 --          {$ENDC}
20 580 --          END;
20 581 --
20 582 --          IF (partCode = inUpButton) OR (partCode = inPageUp) THEN
20 583 --              delta.vh[vhs] := -delta.vh[vhs];
20 584 --              DoScroll(delta, kRedraw);
20 585 --              ScrollStep := delta.vh[vhs];
20 586 -- 0 A  END {TScroller.ScrollStep};
20 587 --
20 588 --      {-----+
20 589 --      |  ScrollTo  |
20 590 --      +-----}
20 591 --      {$S MAScroll}
20 592 -- 0 A  PROCEDURE TScroller.ScrollTo (h, v: VCoordinate; redraw: BOOLEAN);
20 593 --

```

```

20 594 0- A BEGIN
20 595 -- ScrollBy(h - fTranslation.h, v - fTranslation.v, redraw);
20 596 -0 A END {TScroller.ScrollTo};
20 597 --
20 598 -- {-----+
20 599 -- | SetScrollLimits |
20 600 -- +-----}
20 601 -- {$S MANonRes}
20 602 - A PROCEDURE TScroller.SetScrollLimits (scrollLimit: VPoint; drawScrollBars: BOOLEAN);
20 603 --
20 604 -- VAR
20 605 --     maxCoord:      VCoordinate;
20 606 --     vhs:           VHSelect;
20 607 --     newTranslation: VPoint;
20 608 --
20 609 0- A BEGIN
20 610 --     fScrollLimit := scrollLimit;
20 611 --     newTranslation := fTranslation;
20 612 --     FOR vhs := v TO h DO
20 613 1- BEGIN
20 614 --         maxCoord := Max(0, scrollLimit.vh[vhs] - fSize.vh[vhs]);
20 615 --         IF maxCoord <> fMaxTranslation.vh[vhs] THEN
20 616 2- BEGIN
20 617 --             fMaxTranslation.vh[vhs] := maxCoord;
20 618 --             IF fScrollBars[vhs] <> NIL THEN
20 619 --                 fScrollBars[vhs].SetLongMax(maxCoord, drawScrollBars);
20 620 --
20 621 --             IF maxCoord < fTranslation.vh[vhs] THEN
20 622 --                 newTranslation.vh[vhs] := maxCoord;
20 623 -2 END;
20 624 -1 END;
20 625 --
20 626 --     IF (newTranslation.h <> fTranslation.h) | (newTranslation.v <> fTranslation.v) THEN
20 627 1- BEGIN
20 628 --         ScrollTo(newTranslation.h, newTranslation.v, kDontRedraw);
20 629 --         ForceRedraw;
20 630 -1 END;
20 631 -0 A END {TScroller.SetScrollLimits};
20 632 --
20 633 -- {-----+
20 634 -- | SetScrollParameters |
20 635 -- +-----}
20 636 -- {$S MANonRes}
20 637 - A PROCEDURE TScroller.SetScrollParameters (horzUnits, vertUnits: VCoordinate;
20 638 --     horzConstraint, vertConstraint: BOOLEAN);
20 639 --
20 640 0- A BEGIN
20 641 --     fScrollUnit.h := horzUnits;
20 642 --     fScrollUnit.v := vertUnits;
20 643 --     fConstrain[h] := horzConstraint;
20 644 --     fConstrain[v] := vertConstraint;
20 645 -0 A END {TScroller.SetScrollParameters};
20 646 --
20 647 -- {-----+
20 648 -- | SubViewChangedSize |
20 649 -- +-----}
20 650 -- {$S MANonRes}
20 651 - A PROCEDURE TScroller.SubViewChangedSize (theSubView: TView; delta: VPoint); OVERRIDE;
20 652 --
20 653 0- A BEGIN
20 654 --     SetScrollLimits(theSubView.fSize, kRedraw);
20 655 -0 A END {TScroller.SubViewChangedSize};
20 656 --
20 657 -- {-----+
20 658 -- | SuperToLocal |
20 659 -- +-----}
20 660 -- {$S MAREs}
20 661 - A PROCEDURE TScroller.SuperToLocal (VAR thePoint: VPoint); OVERRIDE;
20 662 0- A BEGIN
20 663 --     SubVpt(fLocation, thePoint);
20 664 --     AddVpt(fTranslation, thePoint);
20 665 -0 A END {TScroller.SuperToLocal};
13 3008 -- {$I UMacApp.TControls.p}

```

```

21 1 -- {SP}
21 2 -- {UMacApp.TControls.p}
21 3 -- {Copyright © 1987-1988 by Apple Computer Inc. All rights reserved.}
21 4 --
21 5 -- (*****
21 6 -- (* GLOBAL ROUTINES *)
21 7 -- (*****
21 8 -- {$S Main}
21 9 -- {-----+
21 10 -- | SBarActionProc
21 11 -- +-----}
21 12 -- A PROCEDURE SBarActionProc (aCMgrControl: ControlHandle; partCode: INTEGER);
21 13 -- VAR
21 14 --     aScrollBar: TScrollBar;
21 15 --     backwards: BOOLEAN;
21 16 --
21 17 0- A BEGIN
21 18 --     IF partCode <> 0 THEN
21 19 1- BEGIN
21 20 --         aScrollBar := TScrollBar(GetCRefCon(aCMgrControl));
21 21 --         backwards := (partCode = inUpButton) | (partCode = inPageUp);
21 22 --         WITH aScrollBar DO
21 23 --             IF (backwards & (fLongVal > fLongMin)) |
21 24 --                 ((NOT backwards) & (fLongVal < fLongMax)) THEN
21 25 --                 aScrollBar.TrackScrollBar(partCode);
21 26 --         END;
21 27 0- A END;
21 28 --
21 29 -- (*****
21 30 -- (* TControlTracker *)
21 31 -- (*****
21 32 -- {$S MASelCommand}
21 33 -- {-----+
21 34 -- | IControlTracker
21 35 -- +-----}
21 36 -- A PROCEDURE TControlTracker.IControlTracker (theControl: TControl);
21 37 0- A BEGIN
21 38 --     ICommand(cTrackingControl, NIL, theControl, NIL);
21 39 --     {$IFC qDebug}
21 40 --         IF theControl = NIL THEN
21 41 --             ProgramBreak('Control passed to IControlTracker can't be NIL.');
```



```

21 99 0- A BEGIN
21 100 --      {$IFC qDebug}
21 101 --          IF (itsSize.h > kMaxCoord) | (itsSize.v > kMaxCoord) THEN
21 102 --              ProgramBreak('TControl.IControl: Size is too large.');
```

21 103 -- {\$ENDC}

```

21 104 --
21 105 --          fTextStyle := gSystemStyle;
21 106 --          IView(NIL, itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
21 107 --          fDefChoice := mOKHit;
21 108 --          fHilite := FALSE;
21 109 --          fDimmed := FALSE;
21 110 --          fSizeable := TRUE;
21 111 --          fAdornment := [];
21 112 --          fPenSize := Point($00010001);
21 113 --      {$H-}
21 114 --          fInset := gZeroRect;
21 115 --      {$H+}
21 116 --          fTextStyle := gSystemStyle;
21 117 --          fDismissesDialog := FALSE;
21 118 -0 A END;
21 119 --      {$ENDC}
21 120 --
21 121 --
21 122 --      {$IFC qTemplateViews}
21 123 --      {$SS MAOpen}
21 124 --      {-----+
21 125 --      | IRes
21 126 --      +-----}
21 127 -- A PROCEDURE TControl.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr);
21 128 --
21 129 --      VAR
21 130 --          aTextStyle:    TextStyle;
21 131 --
21 132 0- A BEGIN
21 133 --          { Don't pass the document to INHERITED IRes. This avoids putting the control on the
21 134 --            document's view list.}
21 135 --          INHERITED IRes(NIL, itsSuperView, itsParams);
21 136 --          fDocument := itsDocument;
21 137 --
21 138 --          WITH ControlTemplatePtr(itsParams)^ DO
21 139 1- BEGIN
21 140 --              fDefChoice := mOKHit;          {??? This seems questionable}
21 141 --              fAdornment := CntlAdornment(itsAdornment);
21 142 --              fSizeable := isSizable;
21 143 --              fHilite := isHilited;
21 144 --              fDimmed := isDimmed;
21 145 --              fDismissesDialog := canDismiss;
21 146 --              fInset := itsInset;
21 147 --              fPenSize := itsPenSize;
21 148 --              SetTextStyle(aTextStyle, GetFontNum(itsFontName), itsTextFace, itsTextSize, itsTextColor);
21 149 --              fTextStyle := aTextStyle;
21 150 --
21 151 --              OffsetPtrWStr(itsParams, SIZEOF(ControlTemplate));
21 152 -1 END;
21 153 -0 A END;
21 154 --      {$ENDC}
21 155 --
21 156 --
21 157 --      {$IFC qWriteTemplates}
21 158 --      {$SS MAWriteRes}
21 159 --      {-----+
21 160 --      | WRes
21 161 --      +-----}
21 162 -- A PROCEDURE TControl.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 163 --
21 164 --      VAR
21 165 --          theSize:    INTEGER;
21 166 --          theFont:    Str255;
21 167 --          cnPtr:      ControlTemplatePtr;
21 168 0- A BEGIN
21 169 --          INHERITED WRes(theResource, itsParams);
21 170 --
21 171 --          theSize := fTextStyle.tsSize;
21 172 --          GetPortFontInfo(fTextStyle.tsFont, theFont, theSize);
21 173 --
21 174 --          cnPtr := ControlTemplatePtr(ExpandPtrWStr(theResource, itsParams,
21 175 --            SIZEOF(ControlTemplate), LENGTH(theFont)));
21 176 --
21 177 --          WITH cnPtr^ DO
21 178 1- BEGIN
21 179 --              itsAdornment := SignedByte(fAdornment);
21 180 --              itsPenSize := fPenSize;
21 181 --              isSizable := fSizeable;
21 182 --              isDimmed := fDimmed;
21 183 --              isHilited := fHilite;
21 184 --              canDismiss := fDismissesDialog;
21 185 --              itsInset := fInset;
21 186 --              WITH fTextStyle DO
21 187 2- BEGIN
21 188 --                  itsTextFace := tsFace;
21 189 --                  itsTextSize := theSize;
21 190 --                  itsTextColor := tsColor;
21 191 -2 END;
21 192 --          (* itsFontName := theFont; *)
21 193 --          CopyStr255(theFont, PRStr(itsFontName));
21 194 -1 END;
21 195 -0 A END;
21 196 --      {$ENDC}
21 197 --

```

```

21 198 --
21 199 --      {$IFC qWriteTemplates}
21 200 --      {$S MAWriteRes}
21 201 --      {-----+
21 202 --      |   WriteRes
21 203 --      +-----}
21 204 -- A  PROCEDURE TControl.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 205 0- A  BEGIN
21 206 --      gWResSignature := 'cntl';   gWResType := 'TControl';
21 207 --      WRes(theResource, itsParams);
21 208 -0 A  END;
21 209 --      {$ENDC}
21 210 --
21 211 --
21 212 --      {$S MAActivate}
21 213 --      {-----+
21 214 --      |   Activate
21 215 --      +-----}
21 216 -- A  PROCEDURE TControl.Activate (entering: BOOLEAN); OVERRIDE;
21 217 --
21 218 0- A  BEGIN
21 219 --      ( Don't do anything )
21 220 -0 A  END;
21 221 --
21 222 --      {$S MAREs}
21 223 --      {-----+
21 224 --      |   ComputeSize
21 225 --      +-----}
21 226 -- A  PROCEDURE TControl.ComputeSize (VAR newSize: VPoint); OVERRIDE;
21 227 --      VAR
21 228 --          deltaSz:   Point;
21 229 --
21 230 0- A  BEGIN
21 231 --      INHERITED ComputeSize(newSize);
21 232 --      IF NOT fSizeable THEN
21 233 1-          BEGIN
21 234 --              deltaSz.h := newSize.h - fSize.h;  ( _we need to adjust the bottom/right offset )
21 235 --              deltaSz.v := newSize.v - fSize.v;  ( Controls cannot be bigger than QD space. )
21 236 --
21 237 --              IF LONGINT(deltaSz) <> 0 THEN      ( If view is going to change size, then_ )
21 238 --                  {$H-}                          { _adjust bottom/right offset so it_ }
21 239 --                  AddPt(deltaSz, fInset.botRight);  ( _doesn't change control's actual size )
21 240 --                  {$H+}
21 241 -1          END;
21 242 -0 A  END;
21 243 --
21 244 --      {$S MAREs}
21 245 --      {-----+
21 246 --      |   ContainsMouse
21 247 --      +-----}
21 248 -- A  FUNCTION TControl.ContainsMouse (theMouse: VPoint): BOOLEAN; OVERRIDE;
21 249 --      VAR
21 250 --          aRect:   Rect;
21 251 0- A  BEGIN
21 252 --      ControlArea(aRect);
21 253 --      ContainsMouse := PtInRect(VPtToPt(theMouse), aRect);
21 254 -0 A  END;
21 255 --
21 256 --      {$S MAREs}
21 257 --      {-----+
21 258 --      |   ControlArea
21 259 --      +-----}
21 260 -- A  PROCEDURE TControl.ControlArea (VAR theArea: Rect);
21 261 0- A  BEGIN
21 262 --      {$H-}
21 263 --      WITH fInset DO
21 264 --          SetRect(theArea, left, top, fSize.h - right, fSize.v - bottom);
21 265 --      {$H+}
21 266 --      IF adnShadow IN fAdornment THEN
21 267 --          SubPt(fPenSize, theArea.botRight);
21 268 -0 A  END;
21 269 --
21 270 --      {$S MAREs}
21 271 --      {-----+
21 272 --      |   Dim
21 273 --      +-----}
21 274 -- A  PROCEDURE TControl.Dim;
21 275 --      VAR
21 276 --          area:   Rect;
21 277 0- A  BEGIN
21 278 --      ControlArea(area);
21 279 --      PenPat(gray);
21 280 --      PenMode(patBic);
21 281 --      PaintRect(area);
21 282 -0 A  END;
21 283 --
21 284 --      {$S MANonRes}
21 285 --      {-----+
21 286 --      |   DimState
21 287 --      +-----}
21 288 -- A  PROCEDURE TControl.DimState (state, redraw: BOOLEAN);
21 289 0- A  BEGIN
21 290 --      IF state <> fDimmed THEN
21 291 1-          BEGIN
21 292 --              fDimmed := state;
21 293 --              IF redraw THEN
21 294 --                  DrawContents;
21 295 -1          END;
21 296 -0 A  END;

```

```

21 297 --
21 298 --      ($S MASElCommand)
21 299 --      {-----+
21 300 --      | DoMouseCommand
21 301 --      +-----+
21 302 -- A FUNCTION TControl.DoMouseCommand (VAR theHouse: Point; VAR info: EventInfo;
21 303 --      VAR hysteresis: Point): TCommand; OVERRIDE:
21 304 --      VAR
21 305 --          aControlTracker: TControlTracker;
21 306 0- A BEGIN
21 307 --          NEW(aControlTracker);
21 308 --          FailNIL(aControlTracker);
21 309 --          aControlTracker.IControlTracker(SELF);
21 310 --          DoMouseCommand := aControlTracker;
21 311 -0 A END;
21 312 --
21 313 --      ($S MAREs)
21 314 --      {-----+
21 315 --      | Draw
21 316 --      +-----+
21 317 -- A PROCEDURE TControl.Draw (area: Rect); OVERRIDE:
21 318 --      VAR
21 319 --          qdExtent: Rect;
21 320 --
21 321 0- A BEGIN
21 322 --      IF SignedByte(fAdornment) <> 0 THEN
21 323 1-          BEGIN
21 324 --              GetQDExtent(qdExtent);
21 325 --              Adorn(qdExtent, fPenSize, fAdornment);
21 326 -1          END;
21 327 --      IF fDimmed THEN
21 328 --          Dim;
21 329 --      IF fHilite THEN
21 330 --          Hilite;
21 331 -0 A END;
21 332 --
21 333 --      ($S MAREs)
21 334 --      {-----+
21 335 --      | Focus
21 336 --      +-----+
21 337 -- A FUNCTION TControl.Focus: BOOLEAN; OVERRIDE:
21 338 0- A BEGIN
21 339 --      IF INHERITED Focus THEN
21 340 1-          BEGIN
21 341 --              SetPortTextStyle(fTextStyle);
21 342 --              Focus := TRUE;
21 343 -1          END
21 344 --      ELSE
21 345 --          Focus := FALSE;
21 346 -0 A END;
21 347 --
21 348 --      ($S MAREs)
21 349 --      {-----+
21 350 --      | Hilite
21 351 --      +-----+
21 352 -- A PROCEDURE TControl.Hilite:
21 353 --      VAR
21 354 --          aRect: Rect;
21 355 0- A BEGIN
21 356 --          ControlArea(aRect);
21 357 --          InvertRect(aRect);
21 358 -0 A END;
21 359 --
21 360 --      ($S MANonRes)
21 361 --      {-----+
21 362 --      | HiliteState
21 363 --      +-----+
21 364 -- A PROCEDURE TControl.HiliteState (state, redraw: BOOLEAN);
21 365 0- A BEGIN
21 366 --      IF state <> fHilite THEN
21 367 1-          BEGIN
21 368 --              fHilite := state;
21 369 --              IF redraw & Focus THEN
21 370 --                  Hilite;
21 371 -1          END;
21 372 -0 A END;
21 373 --
21 374 --      ($S MANonRes)
21 375 --      {-----+
21 376 --      | Inset
21 377 --      +-----+
21 378 -- A PROCEDURE TControl.Inset (dh, dv: INTEGER; redraw: BOOLEAN);
21 379 0- A BEGIN
21 380 --      IF fSizeable THEN
21 381 1-          BEGIN
21 382 --              ($H-)
21 383 --              OffsetRect(fInset, dh, dv);
21 384 --              ($H+)
21 385 --              Resize(fSize.h, fSize.v, redraw);
21 386 -1          END;
21 387 -0 A END;
21 388 --
21 389 --      ($S MANonRes)
21 390 --      {-----+
21 391 --      | InstallColor
21 392 --      +-----+
21 393 -- A PROCEDURE TControl.InstallColor(theColor: RGBColor; redraw: BOOLEAN);
21 394 0- A BEGIN
21 395 --      fTextStyle.tsColor := theColor;

```

```

21 396 --      IF redraw THEN
21 397 --          DrawContents:
21 398 -0 A  END;
21 399 --
21 400 --      {$$ MAnonRes}
21 401 --      {-----+
21 402 --      |   InstallTextStyle
21 403 --      +-----+}
21 404 -- A  PROCEDURE TControl.InstallTextStyle (theTextStyle: TextStyle; redraw: BOOLEAN);
21 405 -0 A  BEGIN
21 406 --      fTextStyle := theTextStyle;
21 407 --      IF redraw THEN
21 408 --          DrawContents;
21 409 -0 A  END;
21 410 --
21 411 --      {$$ MRes}
21 412 --      {-----+
21 413 --      |   IsDimmed
21 414 --      +-----+}
21 415 -- A  FUNCTION TControl.IsDimmed: BOOLEAN;
21 416 -0 A  BEGIN
21 417 --      IsDimmed := fDimmed;
21 418 -0 A  END;
21 419 --
21 420 --      {$$ MAnonRes}
21 421 --      {-----+
21 422 --      |   Resize
21 423 --      +-----+}
21 424 -- A  PROCEDURE TControl.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
21 425 --
21 426 -0 A  BEGIN
21 427 --      { If we need to invalidate, then invalidate the whole view rather than
21 428 --        the difference between the old and new size. }
21 429 --      INHERITED Resize(width, height, invalidate);
21 430 --      IF invalidate THEN
21 431 --          ForceRedraw;
21 432 -0 A  END;
21 433 --
21 434 --      {$$ MAnonRes}
21 435 --      {-----+
21 436 --      |   SetInset
21 437 --      +-----+}
21 438 -- A  PROCEDURE TControl.SetInset(newInset: Rect; redraw: BOOLEAN);
21 439 -0 A  BEGIN
21 440 --      IF fsizeable THEN
21 441 -1-      BEGIN
21 442 --          fInset := newInset;
21 443 --          Resize(fSize.h, fSize.v, redraw);
21 444 -1  END;
21 445 -0 A  END;
21 446 --
21 447 --      {$$ MASelCommand}
21 448 --      {-----+
21 449 --      |   TrackConstrain
21 450 --      +-----+}
21 451 -- A  PROCEDURE TControl.TrackConstrain(anchorPoint, previousPoint: VPoint;
21 452 --                                     VAR nextPoint: VPoint);
21 453 -0 A  BEGIN
21 454 -0 A  END;
21 455 --
21 456 --      {$$ MASelCommand}
21 457 --      {-----+
21 458 --      |   TrackFeedback
21 459 --      +-----+}
21 460 -- A  PROCEDURE TControl.TrackFeedback(anchorPoint, nextPoint: VPoint;
21 461 --                                     turnItOn, mouseDidMove: BOOLEAN);
21 462 -0 A  BEGIN
21 463 -0 A  END;
21 464 --
21 465 --      {$$ MASelCommand}
21 466 --      {-----+
21 467 --      |   TrackMouse
21 468 --      +-----+}
21 469 -- A  PROCEDURE TControl.TrackMouse(aTrackPhase: TrackPhase;
21 470 --                                  VAR anchorPoint, previousPoint, nextPoint: VPoint;
21 471 --                                  mouseDidMove: BOOLEAN);
21 472 -0 A  BEGIN
21 473 -1-      CASE aTrackPhase OF
21 474 --          trackPress:
21 475 --              HiliteState(TRUE, kRedraw);
21 476 --          trackMove:
21 477 --              HiliteState(ContainsMouse(nextPoint), kRedraw);
21 478 --          trackRelease:
21 479 -2-      BEGIN
21 480 --              IF fHilite THEN
21 481 --                  HiliteState(FALSE, kRedraw);
21 482 --              IF ContainsMouse(nextPoint) THEN
21 483 --                  DoChoice(SELf, fDefChoice);
21 484 -2  END;
21 485 -1  END;
21 486 -0 A  END;
21 487 --
21 488 --      {$$ MRes}
21 489 --      {-----+
21 490 --      |   Validate
21 491 --      +-----+}
21 492 -- A  FUNCTION TControl.Validate: BOOLEAN;
21 493 -0 A  BEGIN
21 494 --      Validate := TRUE;

```

```

21 495 -- A   END;
21 496 --
21 497 --   {SIFC qDebug}
21 498 --   {SS MAFields}
21 499 --   {-----+
21 500 --   | Fields
21 501 --   +-----}
21 502 -- A   PROCEDURE TControl.Fields (PROCEDURE DoToField (fieldName: Str255;
21 503 --                                     fieldAddr: Ptr;
21 504 --                                     fieldType: INTEGER)); OVERRIDE;
21 505 0- A   BEGIN
21 506 --       DoToField('TControl', NIL, bClass);
21 507 --       DoToField('fDefChoice', @fDefChoice, bInteger);
21 508 --       DoToField('fHilite', @fHilite, bBoolean);
21 509 --       DoToField('fDimmed', @fDimmed, bBoolean);
21 510 --       DoToField('fSizeable', @fSizeable, bBoolean);
21 511 --       DoToField('fDismissesDialog', @fDismissesDialog, bBoolean);
21 512 --       DoToField('fAdornment', @fAdornment, bCntrlAdornment);
21 513 --       DoToField('fPenSize', @fPenSize, bPoint);
21 514 --       DoToField('fInset', @fInset, bRect);
21 515 --       {$H-}
21 516 --       TextStyleFields('fTextStyle', fTextStyle, DoToField);
21 517 --       {$H+}
21 518 --       INHERITED Fields(DoToField);
21 519 -- A   END;
21 520 --   {$ENDC}
21 521 --
21 522 --
21 523 --   (*****
21 524 --   (*   T C t l M g r
21 525 --   (*****
21 526 --
21 527 --   {SIFC qProceduralViews}
21 528 --   {SS MAOpen}
21 529 --   {-----+
21 530 --   | ICtlMgr
21 531 --   +-----}
21 532 -- A   PROCEDURE TCtlMgr.ICtlMgr (itsSuperView: TView; itsLocation, itsSize: VPoint;
21 533 --                                     itsHSizeDet, itsVSizeDet: SizeDeterminer;
21 534 --                                     itsTitle: Str255; itsVal, itsMin, itsMax, itsProcID: INTEGER);
21 535 --   VAR
21 536 --       itsBounds:      Rect;
21 537 --
21 538 0- A   BEGIN
21 539 --       fCmgrControl := NIL;
21 540 --       IControl(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet);
21 541 --       SetRect(itsBounds, 0, 0, fSize.h, fSize.v);
21 542 --       CreateCMgrControl(itsBounds, itsTitle, itsVal, itsMin, itsMax, itsProcID);
21 543 -- A   END;
21 544 --   {$ENDC}
21 545 --
21 546 --
21 547 --   {SIFC qTemplateViews}
21 548 --   {SS MAOpen}
21 549 --   {-----+
21 550 --   | IRes
21 551 --   +-----}
21 552 -- A   PROCEDURE TCtlMgr.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr);
21 553 --
21 554 0- A   BEGIN
21 555 --       fCmgrControl := NIL;
21 556 --       INHERITED IRes(itsDocument, itsSuperView, itsParams);
21 557 --       (The subclass must create the Control Manager control)
21 558 -- A   END (TCtlMgr.IRes);
21 559 --   {$ENDC}
21 560 --
21 561 --
21 562 --   {SIFC qWriteTemplates}
21 563 --   {SS MAWriteRes}
21 564 --   {-----+
21 565 --   | WRes
21 566 --   +-----}
21 567 -- A   PROCEDURE TCtlMgr.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 568 0- A   BEGIN
21 569 --       INHERITED WRes(theResource, itsParams);
21 570 -- A   END;
21 571 --   {$ENDC}
21 572 --
21 573 --
21 574 --   {SIFC qWriteTemplates}
21 575 --   {SS MAWriteRes}
21 576 --   {-----+
21 577 --   | WriteRes
21 578 --   +-----}
21 579 -- A   PROCEDURE TCtlMgr.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 580 0- A   BEGIN
21 581 --       { We don't write TCtlMgrs_ }
21 582 -- A   END;
21 583 --   {$ENDC}
21 584 --
21 585 --
21 586 --   {$S MAClose}
21 587 --   {-----+
21 588 --   | Free
21 589 --   +-----}
21 590 -- A   PROCEDURE TCtlMgr.Free; OVERRIDE;
21 591 --
21 592 0- A   BEGIN
21 593 --       IF fCmgrControl <> NIL THEN

```

```

21 594 --      DisposeControl(fCMgrControl);
21 595 --      INHERITED Free;
21 596 -0 A  END;
21 597 --
21 598 --      {$S MANonRes}
21 599 --      {-----+
21 600 --      |      BeInPort
21 601 --      +-----+}
21 602 -- A  PROCEDURE TctlMgr.BeInPort (itsPort: GrafPtr); OVERRIDE;
21 603 --
21 604 0- A  BEGIN
21 605 --      IF fCMgrControl <> NIL THEN
21 606 --          WITH fCMgrControl^^ DO
21 607 --              IF itsPort = NIL THEN
21 608 1-          BEGIN
21 609 --              contrlVis := ORD(FALSE);      { So Ctrl Mgr doesn't try to draw it }
21 610 --              contrlOwner := WindowPtr(gWorkPort);
21 611 -1          END
21 612 --          ELSE
21 613 1-          BEGIN
21 614 --              contrlVis := ORD(TRUE);
21 615 --              contrlOwner := WindowPtr(itsPort);
21 616 -1          END;
21 617 -0 A  END;
21 618 --
21 619 --      {$S MAOpen}
21 620 --      {-----+
21 621 --      |      CreateCMgrControl
21 622 --      +-----+}
21 623 -- A  PROCEDURE TctlMgr.CreateCMgrControl (itsBounds: Rect; itsTitle: Str255;
21 624 --                                     itsValue, itsMin, itsMax, itsProcID: INTEGER);
21 625 --
21 626 --      VAR
21 627 --          itsPort:      GrafPtr;
21 628 --          aCMgrControl: ControlHandle;
21 629 --          fi:           FailInfo;
21 630 --
21 631 -- B      PROCEDURE HandleFailure(error: OSErr; message: LONGINT);
21 632 0- B  BEGIN
21 633 --          Free;
21 634 -0 B  END;
21 635 --
21 636 0- A  BEGIN
21 637 --      itsPort := GetGrafPort;
21 638 --      IF itsPort = NIL THEN
21 639 --          itsPort := gWorkPort;
21 640 --      CatchFailures(fi, HandleFailure);
21 641 --      aCMgrControl := NewControl(itsPort, itsBounds, itsTitle, FALSE,
21 642 --                               itsValue, itsMin, itsMax, BOR(itsProcID, useWFont),
21 643 --                               LONGINT(SELF));
21 644 --      FailNIL(aCMgrControl);
21 645 --      Success(fi);
21 646 --      WITH WindowPeek(itsPort)^ DO      { Keep control off Window's control list }
21 647 --          contrlList := contrlList^^.nextControl;
21 648 --
21 649 --      aCMgrControl^^.contrlVis := ORD(itsPort <> gWorkPort);
21 650 --      IF fDimmed THEN
21 651 --          aCMgrControl^^.contrlHilite := 255;
21 652 --      fCMgrControl := aCMgrControl;
21 653 -0 A  END (TctlMgr.CreateCMgrControl);
21 654 --
21 655 --      {$S MANonRes}
21 656 --      {-----+
21 657 --      |      DimState
21 658 --      +-----+}
21 659 -- A  PROCEDURE TctlMgr.DimState (state, redraw: BOOLEAN); OVERRIDE;
21 660 --
21 661 -- B      PROCEDURE SetHilite;
21 662 0- B  BEGIN
21 663 --          HiliteControl(fCMgrControl, ORD(state) * 255);
21 664 --          fDimmed := state;
21 665 -0 B  END;
21 666 --
21 667 0- A  BEGIN
21 668 --      IF fDimmed <> state THEN
21 669 --          WhileFocused(SetHilite, redraw);
21 670 -0 A  END;
21 671 --
21 672 --      {$S MASElCommand}
21 673 --      {-----+
21 674 --      |      DoMouseCommand
21 675 --      +-----+}
21 676 -- A  FUNCTION TctlMgr.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
21 677 --                                   VAR hysteresis: Point): TCommand; OVERRIDE;
21 678 --
21 679 0- A  BEGIN
21 680 --      IF TestControl(fCMgrControl, theMouse) <> 0 THEN
21 681 --          IF TrackControl(fCMgrControl, theMouse, NIL) <> 0 THEN
21 682 --              DoChoice(SELf, fDefChoice);
21 683 --              DoMouseCommand := gNoChanges;
21 684 -0 A  END (TctlMgr.DoMouseCommand);
21 685 --
21 686 --      {$S MAREs}
21 687 --      {-----+
21 688 --      |      Draw
21 689 --      +-----+}
21 690 -- A  PROCEDURE TctlMgr.Draw (area: Rect); OVERRIDE;
21 691 --
21 692 --      VAR

```

```

21 693 --      savedOwner:      WindowPtr;
21 694 --      qdExtent:        Rect;
21 695 --
21 696 0- A BEGIN
21 697 --      IF fCMgrControl^.ctrlVis <> 0 THEN
21 698 1-          BEGIN
21 699 --              savedOwner := fCMgrControl^.ctrlOwner;
21 700 --              fCMgrControl^.ctrlOwner := WindowPtr(thePort);
21 701 --
21 702 --              PenNormal; {NECESSARY?}
21 703 --
21 704 --              fCMgrControl^.ctrlVis := 0;          { Force ShowControl to redraw }
21 705 --              ShowControl(fCMgrControl);
21 706 --
21 707 --              fCMgrControl^.ctrlOwner := savedOwner;
21 708 -1          END;
21 709 --
21 710 --      IF SignedByte(fAdornment) <> 0 THEN
21 711 1-          BEGIN
21 712 --              GetQDExtent(qdExtent);
21 713 --              Adorn(qdExtent, fPenSize, fAdornment);
21 714 -1          END;
21 715 -0 A END;
21 716 --
21 717 --      {$S MAREs}
21 718 --      {-----+
21 719 --      |   GetMax
21 720 --      +-----+}
21 721 -- A FUNCTION TCtrlMgr.GetMax: INTEGER;
21 722 --
21 723 0- A BEGIN
21 724 --      GetMax := GetCtlMax(fCMgrControl)
21 725 -0 A END;
21 726 --
21 727 --      {$S MAREs}
21 728 --      {-----+
21 729 --      |   GetMin
21 730 --      +-----+}
21 731 -- A FUNCTION TCtrlMgr.GetMin: INTEGER;
21 732 --
21 733 0- A BEGIN
21 734 --      GetMin := GetCtlMin(fCMgrControl)
21 735 -0 A END;
21 736 --
21 737 --      {$S MAREs}
21 738 --      {-----+
21 739 --      |   GetText
21 740 --      +-----+}
21 741 -- A PROCEDURE TCtrlMgr.GetText (VAR theText: Str255);
21 742 0- A BEGIN
21 743 --      theText := fCMgrControl^.ctrlTitle;
21 744 -0 A END;
21 745 --
21 746 --      {$S MAREs}
21 747 --      {-----+
21 748 --      |   GetVal
21 749 --      +-----+}
21 750 -- A FUNCTION TCtrlMgr.GetVal: INTEGER;
21 751 --
21 752 0- A BEGIN
21 753 --      GetVal := GetCtlValue(fCMgrControl)
21 754 -0 A END;
21 755 --
21 756 --      {$S MANonRes}
21 757 --      {-----+
21 758 --      |   HiliteState
21 759 --      +-----+}
21 760 -- A PROCEDURE TCtrlMgr.HiliteState (state, redraw: BOOLEAN); OVERRIDE;
21 761 --
21 762 -- B      PROCEDURE SetHilite;
21 763 0- B      BEGIN
21 764 --          HiliteControl(fCMgrControl, ORD(state) * 10);
21 765 -0 B      END;
21 766 --
21 767 0- A BEGIN
21 768 --      IF fHilite <> state THEN
21 769 --          WhileFocused(SetHilite, redraw);
21 770 --          fHilite := state;
21 771 -0 A END;
21 772 --
21 773 --      {$S MANonRes}
21 774 --      {-----+
21 775 --      |   Resize
21 776 --      +-----+}
21 777 -- A PROCEDURE TCtrlMgr.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
21 778 --
21 779 -- B      PROCEDURE ResizeCMgrControl;
21 780 --      VAR
21 781 --          aRect:      Rect;
21 782 0- B      BEGIN
21 783 --          SetRect(aRect, fInset.left, fInset.top, width - fInset.right, height - fInset.bottom);
21 784 --          MoveControl(fCMgrControl, aRect.left, aRect.top);
21 785 --          SizeControl(fCMgrControl, ABS(aRect.right-aRect.left), ABS(aRect.bottom-aRect.top));
21 786 -0 B      END;
21 787 --
21 788 0- A BEGIN
21 789 --      IF fSizeable & (fCMgrControl <> NIL) THEN
21 790 --          WhileFocused(ResizeCMgrControl, kDontRedraw);
21 791 --      INHERITED Resize(width, height, invalidate);

```

```

21 792 -O A   END;
21 793 --
21 794 --      {$S MAREs}
21 795 --      {-----+
21 796 --      |   SetMax
21 797 --      +-----}
21 798 -- A   PROCEDURE TctlMgr.SetMax (itsMax: INTEGER; redraw: BOOLEAN);
21 799 --
21 800 --      {-----+
21 801 --      |   DoSetMax
21 802 --      +-----}
21 803 -- B   PROCEDURE DoSetMax;
21 804 0- B   BEGIN
21 805 --       SetCtlMax(fCMgrControl, itsMax);
21 806 -O B   END;
21 807 --
21 808 0- A   BEGIN
21 809 --       WhileFocused(DoSetMax, redraw);
21 810 -O A   END;
21 811 --
21 812 --      {$S MAREs}
21 813 --      {-----+
21 814 --      |   SetMin
21 815 --      +-----}
21 816 -- A   PROCEDURE TctlMgr.SetMin (itsMin: INTEGER; redraw: BOOLEAN);
21 817 --
21 818 --      {-----+
21 819 --      |   DoSetMin
21 820 --      +-----}
21 821 -- B   PROCEDURE DoSetMin;
21 822 0- B   BEGIN
21 823 --       SetCtlMin(fCMgrControl, itsMin);
21 824 -O B   END;
21 825 --
21 826 0- A   BEGIN
21 827 --       WhileFocused(DoSetMin, redraw);
21 828 -O A   END;
21 829 --
21 830 --      {$S MANonRes}
21 831 --      {-----+
21 832 --      |   SetText
21 833 --      +-----}
21 834 -- A   PROCEDURE TctlMgr.SetText (itsText: Str255; redraw: BOOLEAN);
21 835 --
21 836 --      {-----+
21 837 --      |   DoSetText
21 838 --      +-----}
21 839 -- B   PROCEDURE DoSetText;
21 840 0- B   BEGIN
21 841 --       SetCTitle(fCMgrControl, itsText);
21 842 -O B   END;
21 843 --
21 844 0- A   BEGIN
21 845 --       WhileFocused(DoSetText, redraw);
21 846 -O A   END;
21 847 --
21 848 --      {$S MAREs}
21 849 --      {-----+
21 850 --      |   SetVal
21 851 --      +-----}
21 852 -- A   PROCEDURE TctlMgr.SetVal (newVal: INTEGER; redraw: BOOLEAN);
21 853 --
21 854 --      {-----+
21 855 --      |   DoSetVal
21 856 --      +-----}
21 857 -- B   PROCEDURE DoSetVal;
21 858 0- B   BEGIN
21 859 --       SetCtlValue(fCMgrControl, newVal);
21 860 -O B   END;
21 861 --
21 862 0- A   BEGIN
21 863 --       WhileFocused(DoSetVal, redraw);
21 864 -O A   END;
21 865 --
21 866 --      {$S MAREs}
21 867 --      {-----+
21 868 --      |   SetValues
21 869 --      +-----}
21 870 -- A   PROCEDURE TctlMgr.SetValues (itsVal, itsMin, itsMax: INTEGER; redraw: BOOLEAN);
21 871 --
21 872 --      {-----+
21 873 --      |   DoSetValues
21 874 --      +-----}
21 875 -- B   PROCEDURE DoSetValues;
21 876 0- B   BEGIN
21 877 --       SetCtlMin(fCMgrControl, itsMin);
21 878 --       SetCtlMax(fCMgrControl, itsMax);
21 879 --       SetCtlValue(fCMgrControl, itsVal);
21 880 -O B   END;
21 881 --
21 882 0- A   BEGIN
21 883 --       WhileFocused(DoSetValues, redraw);
21 884 -O A   END;
21 885 --
21 886 --      {$S MAREs}
21 887 --      {-----+
21 888 --      |   WhileFocused
21 889 --      +-----}
21 890 -- A   PROCEDURE TctlMgr.WhileFocused (PROCEDURE DoToControl; redraw: BOOLEAN);

```



```

21 891 --
21 892 -- VAR
21 893 --     wasVisible:    Byte;
21 894 --
21 895 0- A BEGIN
21 896 --     IF fCMgrControl <> NIL THEN
21 897 --         IF redraw THEN
21 898 1-             BEGIN
21 899 --                 IF Focus THEN :           { Attempt to focus }
21 900 --                 DoToControl:           { Do it even if we couldn't focus }
21 901 -1                 END
21 902 --             ELSE
21 903 1-                 BEGIN
21 904 --                     wasVisible := fCMgrControl^.contrlVis;
21 905 --                     fCMgrControl^.contrlVis := 0;
21 906 --                     DoToControl;
21 907 --                     fCMgrControl^.contrlVis := wasVisible;
21 908 --                 ($IFC FALSE)
21 909 --                     GetFocus;
21 910 --                     ClipRect(qZeroRect);
21 911 --                     DoToControl;
21 912 --                     SetFocus;
21 913 --                 ($ENDC)
21 914 -1                 END;
21 915 -0 A END;
21 916 --
21 917 -- {$IFC qDebug}
21 918 --     {$S MAFields}
21 919 --     {-----+
21 920 --     | Fields
21 921 --     +-----}
21 922 -- A PROCEDURE TctlMgr.Fields (PROCEDURE DoToField (fieldName: Str255;
21 923 --                             fieldAddr: Ptr;
21 924 --                             fieldType: INTEGER)); OVERRIDE;
21 925 0- A BEGIN
21 926 --     DoToField('TctlMgr', NIL, bClass);
21 927 --     DoToField('fCMgrControl', @fCMgrControl, bControlHandle);
21 928 --     INHERITED Fields(DoToField);
21 929 -0 A END;
21 930 -- {$ENDC}
21 931 --
21 932 -- {*****}
21 933 -- (* T S c r o l l B a r *)
21 934 -- {*****}
21 935 --
21 936 -- {$IFC qProceduralViews}
21 937 -- {$S MAOpen}
21 938 -- {-----+
21 939 -- | IScrollBar
21 940 -- +-----}
21 941 -- A PROCEDURE TScrollBar.IScrollBar (itsSuperView: TView; itsLocation, itsSize: VPoint;
21 942 --                                     itsHSizeDet, itsVSizeDet: SizeDeterminer;
21 943 --                                     itsDirection: VHSelect; itsVal, itsMin, itsMax: LONGINT);
21 944 0- A BEGIN
21 945 --     IctlMgr(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet,
21 946 --             '', itsVal, itsMin, itsMax, scrollBarProc);
21 947 --     fDirection := itsDirection;
21 948 --     SetLongValues(itsVal, itsMin, itsMax, kDontRedraw);
21 949 --     IF itsDirection = h THEN
21 950 --         fDefChoice := mHScrollBarHit
21 951 --     ELSE
21 952 --         fDefChoice := mVScrollBarHit;
21 953 -0 A END;
21 954 -- {$ENDC}
21 955 --
21 956 --
21 957 -- {$IFC qTemplateViews}
21 958 -- {$S MAOpen}
21 959 -- {-----+
21 960 -- | IRes
21 961 -- +-----}
21 962 -- A PROCEDURE TScrollBar.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
21 963 --
21 964 -- VAR
21 965 --     area:          Rect;
21 966 --
21 967 0- A BEGIN
21 968 --     INHERITED IRes(itsDocument, itsSuperView, itsParams);
21 969 --
21 970 --     ControlArea(area);
21 971 --     CreateCMgrControl(area, '', 0, 0, 0, scrollBarProc);
21 972 --     WITH ScrollBarTemplatePtr(itsParams)^ DO
21 973 --         SetLongValues(value, minimum, maximum, kDontRedraw);
21 974 --
21 975 --     WITH area DO
21 976 --         IF (bottom - top) >= (right - left) THEN
21 977 --             fDirection := v
21 978 --         ELSE
21 979 --             fDirection := h;
21 980 --
21 981 --         OffsetPtr(itsParams, SIZEOF(ScrollBarTemplate));
21 982 -0 A END;
21 983 -- {$ENDC}
21 984 --
21 985 --
21 986 -- {$IFC qWriteTemplates}
21 987 -- {$S MAWriteRes}
21 988 -- {-----+
21 989 -- | WRes

```

```

21 990 -- +-----)
21 991 -- A PROCEDURE TScrollBar.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 992 --
21 993 -- VAR
21 994 --     sbPtr: ScrollBarTemplatePtr;
21 995 --
21 996 0- A BEGIN
21 997 --     INHERITED WRes(theResource, itsParams);
21 998 --
21 999 --     sbPtr := ScrollBarTemplatePtr(ExpandPtr(theResource, itsParams,
21 1000 --         SIZEOF(ScrollBarTemplate)));
21 1001 --
21 1002 --     WITH sbPtr^ DO
21 1003 1- BEGIN
21 1004 --         value := fLongVal;
21 1005 --         minimum := fLongMin;
21 1006 --         maximum := fLongMax;
21 1007 -1 END;
21 1008 0- A END;
21 1009 --     ($ENDC)
21 1010 --
21 1011 --
21 1012 --     ($IFC qWriteTemplates)
21 1013 --     ($$ MAWriteRes)
21 1014 --     {-----+
21 1015 --     | WriteRes |
21 1016 --     +-----}
21 1017 -- A PROCEDURE TScrollBar.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 1018 0- A BEGIN
21 1019 --     gWResSignature := 'sbar'; gWResType := 'TScrollBar';
21 1020 --     WRes(theResource, itsParams);
21 1021 0- A END;
21 1022 --     ($ENDC)
21 1023 --
21 1024 --
21 1025 --     ($$ MAScroll)
21 1026 --     {-----+
21 1027 --     | DeltaValue |
21 1028 --     +-----}
21 1029 -- A PROCEDURE TScrollBar.DeltaValue (delta: VCoordinate);
21 1030 --
21 1031 0- A BEGIN
21 1032 --     IF delta <> 0 THEN
21 1033 1- BEGIN
21 1034 --         {Ensure that delta does not cause an overflow (or underflow)}
21 1035 --         IF delta > 0 THEN
21 1036 --             delta := Min(delta, fLongMax - fLongVal)
21 1037 --         ELSE
21 1038 --             delta := Max(delta, fLongMin - fLongVal);
21 1039 --
21 1040 --             SetLongVal(fLongVal + delta, TRUE);
21 1041 -1 END;
21 1042 0- A END;
21 1043 --
21 1044 --     ($$ MAScroll)
21 1045 --     {-----+
21 1046 --     | DoMouseCommand |
21 1047 --     +-----}
21 1048 -- A FUNCTION TScrollBar.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
21 1049 --     VAR hysteresis: Point): TCommand; OVERRIDE;
21 1050 --
21 1051 -- VAR
21 1052 --     partCode: INTEGER;
21 1053 --     oldLongValue: VCoordinate;
21 1054 0- A BEGIN
21 1055 1- CASE TestControl(fCMgrControl, theMouse) OF
21 1056 --     InUpButton, inDownButton, inPageUp, inPageDown:
21 1057 2- BEGIN
21 1058 --         GetFocus;
21 1059 --         partCode := TrackControl(fCMgrControl, theMouse, @SBarActionProc);
21 1060 -2 END;
21 1061 --     inThumb:
21 1062 --         IF TrackControl(fCMgrControl, theMouse, NIL) = inThumb THEN
21 1063 2- BEGIN
21 1064 --             oldLongValue := fLongVal;
21 1065 --             {If thumb is dragged to bottom of scroll bar then ensure that
21 1066 --             the new long value is set to the maximum long value.}
21 1067 --             IF GetVal = GetMax THEN
21 1068 --                 fLongVal := fLongMax
21 1069 --             ELSE
21 1070 --                 fLongVal := BSL(GetVal, fBitsToShift);
21 1071 --                 DeltaValue(fLongVal - oldLongValue);
21 1072 -2 END;
21 1073 -1 END;
21 1074 --         DoMouseCommand := gNoChanges;
21 1075 0- A END;
21 1076 --
21 1077 --     ($$ MRes)
21 1078 --     {-----+
21 1079 --     | GetLongMax |
21 1080 --     +-----}
21 1081 -- A FUNCTION TScrollBar.GetLongMax: VCoordinate;
21 1082 0- A BEGIN
21 1083 --     GetLongMax := fLongMax;
21 1084 0- A END;
21 1085 --
21 1086 --     ($$ MRes)
21 1087 --     {-----+
21 1088 --     | GetLongMin |

```

```

21 1089 -- +-----+
21 1090 -- A FUNCTION TScrollBar.GetLongMin: VCoordinate;
21 1091 0- A BEGIN
21 1092 --     GetLongMin := fLongMin;
21 1093 -0 A END;
21 1094 --
21 1095 -- { $$ MAREs }
21 1096 -- {-----+
21 1097 -- |   GetLongVal   |
21 1098 -- +-----+
21 1099 -- A FUNCTION TScrollBar.GetLongVal: VCoordinate;
21 1100 0- A BEGIN
21 1101 --     GetLongVal := fLongVal;
21 1102 -0 A END;
21 1103 --
21 1104 -- { $$ MAREs }
21 1105 -- {-----+
21 1106 -- |   SetLongMax   |
21 1107 -- +-----+
21 1108 -- A PROCEDURE TScrollBar.SetLongMax (itsMax: VCoordinate; redraw: BOOLEAN);
21 1109 --
21 1110 0- A BEGIN
21 1111 --     IF itsMax <> fLongMax THEN
21 1112 1-         BEGIN
21 1113 --             fLongMax := itsMax;
21 1114 --             fBitsToShift := 0;
21 1115 --             WHILE itsMax > MAXINT DO
21 1116 2-                 BEGIN
21 1117 --                     itsMax := BSR(itsMax, 1);
21 1118 --                     fBitsToShift := fBitsToShift + 1;
21 1119 -2                 END;
21 1120 --             SetMax(itsMax, redraw);
21 1121 -1             END;
21 1122 -0 A END;
21 1123 --
21 1124 -- { $$ MAREs }
21 1125 -- {-----+
21 1126 -- |   SetLongMin   |
21 1127 -- +-----+
21 1128 -- A PROCEDURE TScrollBar.SetLongMin (itsMin: VCoordinate; redraw: BOOLEAN);
21 1129 --
21 1130 0- A BEGIN
21 1131 --     IF itsMin <> fLongMin THEN
21 1132 1-         BEGIN
21 1133 --             fLongMin := itsMin;
21 1134 --             SetMin(BSR(itsMin, fBitsToShift), redraw);
21 1135 -1             END;
21 1136 -0 A END;
21 1137 --
21 1138 -- { $$ MAREs }
21 1139 -- {-----+
21 1140 -- |   SetLongVal   |
21 1141 -- +-----+
21 1142 -- A PROCEDURE TScrollBar.SetLongVal (itsVal: VCoordinate; redraw: BOOLEAN);
21 1143 --
21 1144 0- A BEGIN
21 1145 --     itsVal := Max(fLongMin, Min(itsVal, fLongMax));
21 1146 --     IF itsVal <> fLongVal THEN
21 1147 1-         BEGIN
21 1148 --             fLongVal := itsVal;
21 1149 --             SetVal(BSR(itsVal, fBitsToShift), redraw);
21 1150 -1             END;
21 1151 -0 A END;
21 1152 --
21 1153 -- { $$ MAREs }
21 1154 -- {-----+
21 1155 -- |   SetLongValues |
21 1156 -- +-----+
21 1157 -- A PROCEDURE TScrollBar.SetLongValues (itsVal, itsMin, itsMax: VCoordinate; redraw: BOOLEAN);
21 1158 --
21 1159 0- A BEGIN
21 1160 --     SetLongMax(itsMax, FALSE);
21 1161 --     SetLongMin(itsMin, redraw);
21 1162 --     SetLongVal(itsVal, FALSE);
21 1163 -0 A END;
21 1164 --
21 1165 -- { $$ MAREs }
21 1166 -- {-----+
21 1167 -- |   TrackScrollBar |
21 1168 -- +-----+
21 1169 -- A PROCEDURE TScrollBar.TrackScrollBar (partCode: INTEGER);
21 1170 --
21 1171 0- A BEGIN
21 1172 --     { Does nothing as a default }
21 1173 -0 A END;
21 1174 --
21 1175 -- { $IFC qDebug }
21 1176 -- { $$ MAFields }
21 1177 -- {-----+
21 1178 -- |   Fields   |
21 1179 -- +-----+
21 1180 -- A PROCEDURE TScrollBar.Fields (PROCEDURE DoToField (fieldName: Str255;
21 1181 --                                     fieldAddr: Ptr;
21 1182 --                                     fieldType: INTEGER)); OVERRIDE;
21 1183 --
21 1184 0- A BEGIN
21 1185 --     DoToField('TScrollBar', NIL, bClass);
21 1186 --     DoToField('fBitsToShift', @fBitsToShift, bInteger);
21 1187 --     DoToField('fDirection', @fDirection, bByte);

```

```

21 1188 --      DoToField('fLongVal', @fLongVal, bLongInt);
21 1189 --      DoToField('fLongMin', @fLongMin, bLongInt);
21 1190 --      DoToField('fLongMax', @fLongMax, bLongInt);
21 1191 --      INHERITED Fields(DoToField);
21 1192 -0 A      END;
21 1193 --      ($ENDC)
21 1194 --
21 1195 --
21 1196 --      (*****
21 1197 --      (*      T S S c r o l l B a r      *)
21 1198 --      (*****
21 1199 --
21 1200 --      {SIFC qProceduralViews}
21 1201 --      {$S MAOpen}
21 1202 --      {-----+
21 1203 --      |      ISScrollBar      |
21 1204 --      +-----+
21 1205 -- A      PROCEDURE TSScrollBar.ISScrollBar (itsSuperView: TView; itsLocation, itsSize: VPoint;
21 1206 --      itsHSizeDet, itsVSizeDet: SizeDeterminer;
21 1207 --      itsDirection: VHSelect; itsMax: LONGINT;
21 1208 --      itsScroller: TScroller);
21 1209 --
21 1210 --      VAR
21 1211 --      fi:      FailInfo;
21 1212 --
21 1213 -- B      PROCEDURE HandleFailure(error: OSerr; message: LONGINT);
21 1214 0- B      BEGIN
21 1215 --      Free;
21 1216 -0 B      END;
21 1217 --
21 1218 0- A      BEGIN
21 1219 --      fScrollers := NIL;
21 1220 --      IScrollBar(itsSuperView, itsLocation, itsSize, itsHSizeDet, itsVSizeDet,
21 1221 --      itsDirection, 0, 0, itsMax);
21 1222 --      fCmgrControl^.ctrlVis := 0;      { Start out assuming it's inactive. }
21 1223 --
21 1224 --      CatchFailures(fi, HandleFailure);
21 1225 --      fScrollers := NewList;
21 1226 --      {SIFC qDebug}
21 1227 --      fScrollers.SetEltType('TScroller');
21 1228 --      ($ENDC)
21 1229 --      IF itsScroller <> NIL THEN
21 1230 1-      BEGIN
21 1231 --      fScrollers.InsertLast(itsScroller);
21 1232 --      itsScroller.HaveScrollBar(SELF, fDirection);
21 1233 -1      END;
21 1234 --      Success(fi);
21 1235 -0 A      END;
21 1236 --      ($ENDC)
21 1237 --
21 1238 --
21 1239 --      {SIFC qTemplateViews}
21 1240 --      {$S MAOpen}
21 1241 --      {-----+
21 1242 --      |      IRes      |
21 1243 --      +-----+
21 1244 -- A      PROCEDURE TSScrollBar.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
21 1245 --
21 1246 --      VAR
21 1247 --      fi:      FailInfo;
21 1248 --
21 1249 -- B      PROCEDURE HandleFailure(error: OSerr; message: LONGINT);
21 1250 0- B      BEGIN
21 1251 --      Free;
21 1252 -0 B      END;
21 1253 --
21 1254 0- A      BEGIN
21 1255 --      INHERITED IRes(itsDocument, itsSuperView, itsParams);
21 1256 --      fCmgrControl^.ctrlVis := 0;      { Start out assuming it's inactive. }
21 1257 --
21 1258 --      CatchFailures(fi, HandleFailure);
21 1259 --      fScrollers := NewList;
21 1260 --      {SIFC qDebug}
21 1261 --      fScrollers.SetEltType('TScroller');
21 1262 --      ($ENDC)
21 1263 --      Success(fi);
21 1264 -0 A      END;
21 1265 --      ($ENDC)
21 1266 --
21 1267 --
21 1268 --      {SIFC qWriteTemplates}
21 1269 --      {$S MAWriteRes}
21 1270 --      {-----+
21 1271 --      |      WRes      |
21 1272 --      +-----+
21 1273 -- A      PROCEDURE TSScrollBar.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 1274 0- A      BEGIN
21 1275 --      INHERITED WRes(theResource, itsParams);
21 1276 -0 A      END;
21 1277 --      ($ENDC)
21 1278 --
21 1279 --
21 1280 --      {SIFC qWriteTemplates}
21 1281 --      {$S MAWriteRes}
21 1282 --      {-----+
21 1283 --      |      WriteRes      |
21 1284 --      +-----+
21 1285 -- A      PROCEDURE TSScrollBar.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
21 1286 0- A      BEGIN

```

```

21 1287 --      { We don't write TSScrollBars_ }
21 1288 -- A   END:
21 1289 --      {$ENDC}
21 1290 --
21 1291 --
21 1292 --      {$$ MAClose}
21 1293 --      {-----+
21 1294 --      |   Free
21 1295 --      +-----}
21 1296 -- A   PROCEDURE TSScrollBar.Free; OVERRIDE;
21 1297 --
21 1298 --      {-----+
21 1299 --      |   DetachFromScroller
21 1300 --      +-----}
21 1301 -- B   PROCEDURE DetachFromScroller (theScroller: TScroller);
21 1302 --
21 1303 -- B   BEGIN
21 1304 --       IF theScroller.fScrollBars[fDirection] = SELF THEN
21 1305 --           theScroller.HaveScrollBar(NIL, fDirection);
21 1306 -- B   END;
21 1307 --
21 1308 -- A   BEGIN
21 1309 --       IF fScrollers <> NIL THEN
21 1310 --           fScrollers.Each(DetachFromScroller);
21 1311 --       FreeObject(fScrollers);
21 1312 --       INHERITED Free;
21 1313 -- A   END;
21 1314 --
21 1315 --      {$$ MAActivate}
21 1316 --      {-----+
21 1317 --      |   Activate
21 1318 --      +-----}
21 1319 -- A   PROCEDURE TSScrollBar.Activate (entering: BOOLEAN); OVERRIDE;
21 1320 --
21 1321 -- B   PROCEDURE SetScrollBar;
21 1322 -- B   BEGIN
21 1323 --       IF entering THEN
21 1324 --           ShowControl(fCmgrControl)
21 1325 --       ELSE
21 1326 --           BEGIN
21 1327 --               HideControl(fCmgrControl);
21 1328 --               Draw(fCmgrControl^^.contrlRect);    { Get the frame drawn. }
21 1329 --           END;
21 1330 --       ValidRect(fCmgrControl^^.contrlRect);    { Because Control Manager invalidates it. }
21 1331 -- B   END;
21 1332 --
21 1333 -- A   BEGIN
21 1334 --       WhileFocused(SetScrollBar, kRedraw);
21 1335 -- A   END;
21 1336 --
21 1337 --      {$$ MAScroll}
21 1338 --      {-----+
21 1339 --      |   DoMouseCommand
21 1340 --      +-----}
21 1341 -- A   FUNCTION TSScrollBar.DoMouseCommand (VAR theMouse: Point; VAR info: EventInfo;
21 1342 --                                         VAR hysteresis: Point): TCommand; OVERRIDE;
21 1343 --
21 1344 -- VAR
21 1345 --     partCode:    INTEGER;
21 1346 --     sBarDelta:   LONGINT;
21 1347 --
21 1348 --     {-----+
21 1349 --     |   ScrollToThumb
21 1350 --     +-----}
21 1350 -- B   PROCEDURE ScrollToThumb (theView: TScroller);
21 1351 --
21 1352 -- B   BEGIN
21 1353 --       sBarDelta := sBarDelta + theView.ScrollRelative(fDirection, fLongVal);
21 1354 -- B   END;
21 1355 --
21 1356 -- A   BEGIN
21 1357 --       IF TestControl(fCmgrControl, theMouse) = inThumb THEN
21 1358 --           BEGIN
21 1359 --               IF TrackControl(fCmgrControl, theMouse, NIL) = inThumb THEN
21 1360 --                   BEGIN
21 1361 --                       GetFocus;
21 1362 --                       sBarDelta := 0;
21 1363 --                       {If thumb is dragged to bottom of scroll bar then ensure that
21 1364 --                        the new long value is set to the maximum long value.}
21 1365 --                       IF GetVal = GetMax THEN
21 1366 --                           fLongVal := fLongMax
21 1367 --                       ELSE
21 1368 --                           fLongVal := BSL(GetVal, fBitsToShift);
21 1369 --                       fScrollers.Each(ScrollToThumb);
21 1370 --                       SetFocus;
21 1371 --                       IF sBarDelta <> 0 THEN
21 1372 --                           SetLongVal(fLongVal + sBarDelta, TRUE);
21 1373 --                       END;
21 1374 --                       DoMouseCommand := gNoChanges;
21 1375 --                   END
21 1376 --               ELSE
21 1377 --                   DoMouseCommand := INHERITED DoMouseCommand(theMouse, info, hysteresis);
21 1378 --           END;
21 1379 -- A   END;
21 1380 --
21 1381 --      {$$ MAREs}
21 1382 --      {-----+
21 1383 --      |   Draw
21 1384 --      +-----}
21 1384 -- A   PROCEDURE TSScrollBar.Draw (area: Rect); OVERRIDE;
21 1385 --

```

```

21 1386 --      VAR
21 1387 --          itsRect:      Rect;
21 1388 --
21 1389 0- A      BEGIN
21 1390 --          IF fCMgrControl^.contrlVis = 0 THEN
21 1391 1-              BEGIN
21 1392 --                  PenNormal;                { Make sure we draw solid, 1-pixel wide lines }
21 1393 --                  itsRect := fCMgrControl^.contrlRect;
21 1394 --                  FrameRect(itsRect);
21 1395 -1              END;
21 1396 --
21 1397 --          INHERITED Draw(area);
21 1398 -0 A      END;
21 1399 --
21 1400 --          {$S MAScroll}
21 1401 --          {-----+
21 1402 --          |   TrackScrollBar
21 1403 --          +-----}
21 1404 -- A      PROCEDURE TSScrollBar.TrackScrollBar (partCode: INTEGER);
21 1405 --
21 1406 --      VAR
21 1407 --          sBarDelta:      LONGINT;
21 1408 --
21 1409 --          {-----+
21 1410 --          |   ScrollView
21 1411 --          +-----}
21 1412 -- B      PROCEDURE ScrollView (theView: TScroller);
21 1413 --
21 1414 0- B      BEGIN
21 1415 --          sBarDelta := sBarDelta + theView.ScrollStep(fDirection, partCode);
21 1416 -0 B      END;
21 1417 --
21 1418 0- A      BEGIN
21 1419 --          sBarDelta := 0;
21 1420 --          fScrollers.Each(ScrollView);
21 1421 --          SetFocus;
21 1422 --          DeltaValue(sBarDelta);
21 1423 -0 A      END;
21 1424 --
21 1425 --          {$IFC qDebug}
21 1426 --          {$S MAFields}
21 1427 --          {-----+
21 1428 --          |   Fields
21 1429 --          +-----}
21 1430 -- A      PROCEDURE TSScrollBar.Fields (PROCEDURE DoToField (fieldName: Str255;
21 1431 --                                     fieldAddr: Ptr;
21 1432 --                                     fieldType: INTEGER)); OVERRIDE;
21 1433 --
21 1434 0- A      BEGIN
21 1435 --          DoToField('TSScrollBar', NIL, bClass);
21 1436 --          DoToField('fScrollers', @fScrollers, bObject);
21 1437 --          INHERITED Fields(DoToField);
21 1438 -0 A      END;
21 1439 --          {$ENDC}
13 3009 --      {$I UMacApp.TCommand.p}

```

```

22 1 -- ($P)
22 2 -- { UMacApp.TCommand.p }
22 3 -- { Copyright © 1985-1988 by Apple Computer, Inc. All rights reserved. }
22 4 --
22 5 -- (*****
22 6 -- (*      T C o m m a n d      *)
22 7 -- (*****
22 8 -- ($S MASElCommand)
22 9 -- {-----+
22 10 -- | ICommand |
22 11 -- +-----}
22 12 -- A PROCEDURE TCommand.ICommand(itsCmdNumber: CmdNumber; itsDocument: TDocument;
22 13 --                               itsView: TView; itsScroller: TScroller);
22 14 0- A BEGIN
22 15 --     fCmdNumber := itsCmdNumber;
22 16 --     fChangedDocument := itsDocument;
22 17 --     fTarget := NIL;
22 18 --     fCmdDone := FALSE;
22 19 --     fCanUndo := TRUE;
22 20 --     fCausesChange := TRUE;
22 21 --     fChangesClipboard := FALSE;
22 22 --     fConstrainsMouse := FALSE;
22 23 --     fViewConstrain := TRUE;
22 24 --     fTrackNonMovement := FALSE;
22 25 --     fView := itsView;
22 26 --     fScroller := itsScroller;
22 27 -0 A END;
22 28 --
22 29 --
22 30 -- ($S MADoCommand)
22 31 -- {-----+
22 32 -- | AutoScroll |
22 33 -- +-----}
22 34 -- A PROCEDURE TCommand.AutoScroll (deltaH, deltaV: VCoordinate);
22 35 0- A BEGIN
22 36 --     fScroller.ScrollBy(deltaH, deltaV, kRedraw);
22 37 -0 A END;
22 38 --
22 39 --
22 40 -- ($S MADoCommand)
22 41 -- {-----+
22 42 -- | Commit |
22 43 -- +-----}
22 44 -- A PROCEDURE TCommand.Commit;
22 45 0- A BEGIN
22 46 -0 A END;
22 47 --
22 48 --
22 49 -- ($S MADoCommand)
22 50 -- {-----+
22 51 -- | DoIt |
22 52 -- +-----}
22 53 -- A PROCEDURE TCommand.DoIt;
22 54 0- A BEGIN
22 55 -0 A END;
22 56 --
22 57 --
22 58 -- ($S MADoCommand)
22 59 -- {-----+
22 60 -- | RedoIt |
22 61 -- +-----}
22 62 -- A PROCEDURE TCommand.RedoIt;
22 63 0- A BEGIN
22 64 -0 A END;
22 65 --
22 66 --
22 67 -- ($S MADoCommand)
22 68 -- {-----+
22 69 -- | TrackConstrain |
22 70 -- +-----}
22 71 -- A PROCEDURE TCommand.TrackConstrain(anchorPoint, previousPoint: VPoint;
22 72 --                               VAR nextPoint: VPoint);
22 73 0- A BEGIN
22 74 -0 A END;
22 75 --
22 76 --
22 77 -- ($S MADoCommand)
22 78 -- {-----+
22 79 -- | TrackFeedback |
22 80 -- +-----}
22 81 -- A PROCEDURE TCommand.TrackFeedback(anchorPoint, nextPoint: VPoint;
22 82 --                               turnItOn, mouseDidMove: BOOLEAN);
22 83 --     VAR r: Rect;
22 84 0- A BEGIN
22 85 --     IF mouseDidMove THEN
22 86 1- BEGIN
22 87 --         Pt2Rect(fView.ViewToQDPt(anchorPoint), fView.ViewToQDPt(nextPoint), r);
22 88 --         FrameRect(r);
22 89 -1 END;
22 90 -0 A END;
22 91 --
22 92 --
22 93 -- ($S MADoCommand)
22 94 -- {-----+
22 95 -- | TrackMouse |
22 96 -- +-----}
22 97 -- A FUNCTION TCommand.TrackMouse(aTrackPhase: TrackPhase;
22 98 --                               VAR anchorPoint, previousPoint, nextPoint: VPoint;

```

```

22 99 --          mouseDidMove: BOOLEAN): TCommand;
22 100 0- A BEGIN
22 101 --          TrackMouse := SELF;
22 102 -0 A END;
22 103 --
22 104 --
22 105 --          {$S MAdoCommand}
22 106 --          {-----+
22 107 --          | UndoIt |
22 108 --          +-----}
22 109 -- A PROCEDURE TCommand.UndoIt;
22 110 0- A BEGIN
22 111 -0 A END;
22 112 --
22 113 --
22 114 --          {$IFC qDebug}
22 115 --          {$S MAFields}
22 116 --          {-----+
22 117 --          | Fields |
22 118 --          +-----}
22 119 -- A PROCEDURE TCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
22 120 --                                fieldAddr: Ptr;
22 121 --                                fieldType: INTEGER)); OVERRIDE;
22 122 --
22 123 0- A BEGIN
22 124 --          DoToField('TCommand', NIL, bClass);
22 125 --          DoToField('fCmdNumber', @fCmdNumber, bCmdNumber);
22 126 --          DoToField('fChangedDocument', @fChangedDocument, bObject);
22 127 --          DoToField('fTarget', @fTarget, bObject);
22 128 --          DoToField('fCmdDone', @fCmdDone, bBoolean);
22 129 --          DoToField('fCanUndo', @fCanUndo, bBoolean);
22 130 --          DoToField('fCausesChange', @fCausesChange, bBoolean);
22 131 --          DoToField('fChangesClipboard', @fChangesClipboard, bBoolean);
22 132 --          DoToField('fView', @fView, bObject);
22 133 --          DoToField('fScroller', @fScroller, bObject);
22 134 --          DoToField('fConstrainsMouse', @fConstrainsMouse, bBoolean);
22 135 --          DoToField('fViewConstrain', @fViewConstrain, bBoolean);
22 136 --          INHERITED Fields(DoToField);
22 137 -0 A END ;
22 138 --          {$ENDC}
13 3010 --          {$I UMacApp.TDeskScrapView.p}

```



```

23 1 --      {$P}
23 2 --      { UMacApp.TDeskScrapView.p }
23 3 --      { Copyright © 1985-1988 Apple Computer, Inc. All rights reserved. }
23 4 --
23 5 --
23 6 --      (*****
23 7 --      (*      T D e s k S c r a p V i e w      *)
23 8 --      (*****
23 9 --
23 10 --      {$IFC qProceduralViews}
23 11 --      {$S MAINit}
23 12 --      {-----+
23 13 --      |      IDeskScrapView
23 14 --      +-----}
23 15 -- A  PROCEDURE TDeskScrapView.IDeskScrapView;
23 16 --      VAR
23 17 --          itsLocation:    VPoint;
23 18 --          itsSize:        VPoint;
23 19 --          aPtr:           PScrapStuff;
23 20 0- A  BEGIN
23 21 --          SetVPt(itsLocation, 0, 0);
23 22 --          SetVPt(itsSize, 200, 200);
23 23 --          { Can't know the extent required until we actually read the data in, which we won't bother
23 24 --            to do unless forced to by receiving an Update event }
23 25 --
23 26 --          IView(NIL, NIL, itsLocation, itsSize, sizeVariable, sizeVariable);
23 27 --
23 28 --          aPtr := InfoScrap;
23 29 --          fScrapCount := aPtr^.scrapCount - 1;
23 30 --          fDataHandle := NIL;
23 31 --          fHavePicture := FALSE;
23 32 --          fHaveText := FALSE;
23 33 -0 A  END;
23 34 --      {$ENDC}
23 35 --
23 36 --
23 37 --      {$IFC qTemplateViews}
23 38 --      {$S MAINit}
23 39 --      {-----+
23 40 --      |      IRes
23 41 --      +-----}
23 42 -- A  PROCEDURE TDeskScrapView.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
23 43 --
23 44 --      VAR
23 45 --          aPtr:           PScrapStuff;
23 46 --
23 47 0- A  BEGIN
23 48 --          INHERITED IRes(itsDocument, itsSuperView, itsParams);
23 49 --
23 50 --          aPtr := InfoScrap;
23 51 --          fScrapCount := aPtr^.scrapCount - 1;
23 52 --          fDataHandle := NIL;
23 53 --          fHavePicture := FALSE;
23 54 --          fHaveText := FALSE;
23 55 -0 A  END;
23 56 --      {$ENDC}
23 57 --
23 58 --
23 59 --      {$S MACclipboard}
23 60 --      {-----+
23 61 --      |      Free
23 62 --      +-----}
23 63 -- A  PROCEDURE TDeskScrapView.Free; OVERRIDE;
23 64 --
23 65 0- A  BEGIN
23 66 --          { Never want to free gClipOrphanage. What if some creates another view of this type? }
23 67 -0 A  END;
23 68 --
23 69 --
23 70 --      {$S MACclipboard}
23 71 --      {-----+
23 72 --      |      CalcMinSize
23 73 --      +-----}
23 74 -- A  PROCEDURE TDeskScrapView.CalcMinSize (VAR minSize: VPoint); OVERRIDE;
23 75 --
23 76 --      VAR
23 77 --          vhs:           VHSelect;
23 78 --          aRect:         Rect;
23 79 --          aTEHandle:     TEHandle;
23 80 --          oldHText:      Handle;
23 81 --
23 82 0- A  BEGIN
23 83 --          minSize := fSize;
23 84 --          IF fDataHandle <> NIL THEN
23 85 --              IF fHaveText THEN
23 86 1-          BEGIN { TEXT }
23 87 --              IF Focus THEN
23 88 2-          BEGIN
23 89 --              SetPortTextStyle(gSystemStyle);
23 90 --              fSuperView.GetQDExtent(aRect);
23 91 --              aTEHandle := TENew(aRect, aRect);
23 92 --              FailNIL(aTEHandle);
23 93 --
23 94 --              WITH aTEHandle^^ DO
23 95 3-          BEGIN
23 96 --              oldHText := hText; { We'll need this for TEDispose }
23 97 --              hText := fDataHandle;
23 98 -3  END;

```

```

23 99 --          TEdCallText(aTEHandle);
23 100 --          WITH aTEHandle^^ DO
23 101 3-          BEGIN
23 102 --              minSize.h := aRect.right - aRect.left;
23 103 --              minSize.v := Min(IntMultiply(nLines, lineHeight), kMaxCoord);
23 104 --              hText := oldHText; { So TEdDispose doesn't dispose our handle }
23 105 -3          END;
23 106 --
23 107 --          TEdDispose(aTEHandle);
23 108 -2          END;
23 109 -1          END
23 110 --
23 111 --          ELSE
23 112 1-          BEGIN
23 113 --              {$IFC qDebug}
23 114 --                  IF gIntenseDebugging THEN
23 115 2-                  BEGIN
23 116 --                      Write('Picture Frame: ');
23 117 --                      WriteRect(PicHandle(fDataHandle)^^.picFrame);
23 118 -2                  END;
23 119 --              {$ENDC}
23 120 --
23 121 --              FOR vhs := v TO h DO
23 122 --                  IF NOT gVarClipPicSize THEN
23 123 --                      WITH PicHandle(fDataHandle)^^.picFrame DO
23 124 --                          minSize.vh[vhs] := botRight.vh[vhs] - topLeft.vh[vhs];
23 125 -1          END { PICT };
23 126 -0 A  END;
23 127 --
23 128 --
23 129 --          {$IFC qDebug}
23 130 --          {$S MAFields}
23 131 --          {-----+
23 132 --          | Fields
23 133 --          +-----}
23 134 -- A  PROCEDURE TDeskScrapView.Fields (PROCEDURE DoToField (fieldName: Str255;
23 135 --                                  fieldAddr: Ptr;
23 136 --                                  fieldType: INTEGER));
23 137 --
23 138 0- A  BEGIN
23 139 --          DoToField('TDeskScrapView', NIL, bClass);
23 140 --          DoToField('fHavePicture', @fHavePicture, bBoolean);
23 141 --          DoToField('fHaveText', @fHaveText, bBoolean);
23 142 --          DoToField('fScrapCount', @fScrapCount, bInteger);
23 143 --          DoToField('fDataHandle', @fDataHandle, bHandle);
23 144 --          INHERITED Fields(DoToField);
23 145 -0 A  END;
23 146 --          {$ENDC}
23 147 --
23 148 --
23 149 --          {$S MAClipboard}
23 150 --          {-----+
23 151 --          | CheckScrapContents
23 152 --          +-----}
23 153 -- A  PROCEDURE TDeskScrapView.CheckScrapContents;
23 154 --          VAR
23 155 --              length:      LONGINT;
23 156 --              offset:      LONGINT;
23 157 --              aHandle:      Handle;
23 158 --              resTypeFound: ResType;
23 159 --              savedPerm:    BOOLEAN;
23 160 --              fi:           FailInfo;
23 161 --
23 162 --          {-----+
23 163 --          | Hd1Failure
23 164 --          +-----}
23 165 -- B  PROCEDURE Hd1Failure(error: OSErr; message: LONGINT);
23 166 0- B  BEGIN
23 167 --          IF aHandle <> NIL THEN
23 168 --              DisposHandle(aHandle);
23 169 --              fDataHandle := NIL;
23 170 -0 B  END;
23 171 --
23 172 --          {-----+
23 173 --          | LookForScrapType
23 174 --          +-----}
23 175 -- B  FUNCTION LookForScrapType(theResType: ResType): BOOLEAN;
23 176 --          VAR
23 177 --              err:          LONGINT;
23 178 --              offset:      LongInt;
23 179 0- B  BEGIN
23 180 --              err := GetScrap(NIL, theResType, offset);
23 181 --              IF err > 0 THEN
23 182 --                  resTypeFound := theResType;
23 183 --              {$IFC qDebug}
23 184 --                  IF (err <> noTypeErr) AND (err < 0) THEN
23 185 --                      WriteLn('LookForScrapType: err = ', err:1);
23 186 --              {$ENDC}
23 187 --              LookForScrapType := err > 0;
23 188 -0 B  END;
23 189 --
23 190 0- A  BEGIN
23 191 --          IF fScrapCount <> gNewScrapStuff.scrapCount THEN
23 192 1-          BEGIN
23 193 --
23 194 --              { Make sure we get rid of the old scrap handle if necessary }
23 195 --              IF fDataHandle <> NIL THEN
23 196 2-              BEGIN
23 197 --                  DisposHandle(fDataHandle);

```

```

23 198 --      fDataHandle := NIL;
23 199 -2      END;
23 200 --
23 201 --      fHaveText := LookForScrapType('TEXT');
23 202 --      fHavePicture := LookForScrapType('PICT');
23 203 --
23 204 --      IF fHavePicture OR fHaveText THEN
23 205 2-      BEGIN
23 206 --          aHandle := NewPermHandle(0);
23 207 --          FailNIL(aHandle);
23 208 --
23 209 --          CatchFailures(fi, HdlFailure);
23 210 --          savedPerm := PermAllocation(TRUE);
23 211 --          length := GetScrap(aHandle, resTypeFound, offset);
23 212 --          savedPerm := PermAllocation(savedPerm);
23 213 --
23 214 --          IF length < 0 THEN      { Only results < 0 are really errors }
23 215 --              FailOSErr(length);
23 216 --              fDataHandle := aHandle;
23 217 --              AdjustSize;
23 218 --              IF Focus THEN ;      { Make sure we're still focused }
23 219 --                  Success(fi);
23 220 -2      END;
23 221 -1      END;
23 222 -0 A  END;
23 223 --
23 224 --
23 225 --      {$$ MACclipboard}
23 226 --      {-----+
23 227 --      | Draw |
23 228 --      +-----}
23 229 -- A  PROCEDURE TDeskScrapView.Draw(area: Rect);
23 230 --
23 231 --      VAR
23 232 --          aRect:      Rect;
23 233 --          vhs:        VHSelect;
23 234 --
23 235 0- A  BEGIN
23 236 --
23 237 --          { Has scrap been updated since we last drew? }
23 238 --          CheckScrapContents;
23 239 --
23 240 --          IF fHaveText THEN
23 241 1-      BEGIN
23 242 --              SetPortTextStyle(gSystemStyle);
23 243 --              GetQDExtent(aRect);
23 244 --              { Does TextEdit handle failure in TextBox??? }
23 245 --              TextBox(fDataHandle, GetHandleSize(fDataHandle), aRect, teJustLeft);
23 246 -1      END
23 247 --
23 248 --          ELSE
23 249 --
23 250 --          IF fHavePicture THEN
23 251 1-      BEGIN
23 252 --              IF gVarClipPicSize THEN
23 253 --                  GetQDExtent(aRect)
23 254 --              ELSE
23 255 2-      BEGIN
23 256 --                  aRect := PicHandle(fDataHandle)^^.picFrame;
23 257 --                  FOR vhs := v TO h DO
23 258 3-      BEGIN
23 259 --                  aRect.botRight.vh[vhs] := aRect.botRight.vh[vhs] - aRect.topLeft.vh[vhs];
23 260 --                  aRect.topLeft.vh[vhs] := 0;
23 261 -3      END;
23 262 -2      END;
23 263 --                  DrawPicture(PicHandle(fDataHandle), aRect);
23 264 -1      END;
23 265 --
23 266 --          IF fScrapCount <> gNewScrapStuff.scrapCount THEN
23 267 1-      BEGIN
23 268 --              ValidRect(aRect); { avoid flicker upon clipboard installation -- but could this
23 269 --                                  cause failure to update frame border/appendages sometimes, if that
23 270 --                                  code ever checked for visibility before drawing ??? }
23 271 --              fScrapCount := gNewScrapStuff.scrapCount;
23 272 --              gClipWrittenToDeskScrap := TRUE; { Don't need to write it back out. }
23 273 -1      END;
23 274 -0 A  END;
23 275 --
23 276 --
23 277 --      {$IFC qInspector}
23 278 --      {$$ MAInspector}
23 279 --      {-----+
23 280 --      | GetInspectorName |
23 281 --      +-----}
23 282 -- A  PROCEDURE TDeskScrapView.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
23 283 --
23 284 0- A  BEGIN
23 285 --          IF SELF = gClipView THEN
23 286 --              inspectorName := 'gClipView'
23 287 --          ELSE IF SELF = gClipOrphanage THEN
23 288 --              inspectorName := 'gClipOrphanage';
23 289 -0 A  END;
23 290 --      {$ENDC}
23 291 --
23 292 --
23 293 --      {$$ MACclipboard}
23 294 --      {-----+
23 295 --      | SuperViewChangedSize |
23 296 --      +-----}

```

```
23 297 -- A  PROCEDURE TDeskScrapView.SuperViewChangedSize (delta: VPoint; invalidate: BOOLEAN); OVERRIDE;
23 298 --
23 299 0- A  BEGIN
23 300 --      AdjustSize;
23 301 --      ForceRedraw;
23 302 -0 A  END;
23 303 --
23 304 --
23 305 --      ($$ MAClipboard)
23 306 --      {-----+
23 307 --      |   WriteToDeskScrap   |
23 308 --      +-----}
23 309 -- A  PROCEDURE TDeskScrapView.WriteToDeskScrap; OVERRIDE;
23 310 --
23 311 0- A  BEGIN
23 312 --      { This view represents data that's already written to the desk scrap }
23 313 -0 A  END;
23 314 --
23 315 --
13 3011 --      {$I UMacApp.TPrintHandler.p}
```

```

24 1 --      {$P}
24 2 --      { UMacApp.TPrintHandler.p }
24 3 --      { Copyright © 1985-1988 by Apple Computer, Inc. All rights reserved. }
24 4 --
24 5 --      (*****
24 6 --      (*      T P r i n t H a n d l e r      *)
24 7 --      (*****
24 8 --
24 9 --      {$S MAMain}
24 10 --      {-----+
24 11 --      |      IPrintHandler      |
24 12 --      +-----}
24 13 -- A  PROCEDURE TPrintHandler.IPrintHandler(itsView: TView);
24 14 0- A  BEGIN
24 15 --      fview := itsView;
24 16 --      IEvtHandler(NIL);
24 17 -0 A  END;
24 18 --
24 19 --
24 20 --      {$S MANever}
24 21 --      {-----+
24 22 --      |      BreakFollowing      |
24 23 --      +-----}
24 24 -- A  FUNCTION TPrintHandler.BreakFollowing(vhs: VHSelect; prevBreak: VCoordinate;
24 25 --      VAR automatic: BOOLEAN): VCoordinate;
24 26 0- A  BEGIN
24 27 --      {$IFC qDebug}
24 28 --      ProgramBreak('BreakFollowing called for non-subclassed Printheadler');
24 29 --      {$ENDC}
24 30 -0 A  END;
24 31 --
24 32 --
24 33 --      {$IFC qDebug}
24 34 --      {$S MAFields}
24 35 --      {-----+
24 36 --      |      Fields      |
24 37 --      +-----}
24 38 -- A  PROCEDURE TPrintHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
24 39 --      fieldAddr: Ptr;
24 40 --      fieldType: INTEGER));
24 41 0- A  BEGIN
24 42 --      DoToField('TPrintHandler', NIL, bClass);
24 43 --      DoToField('fView', @fview, bObject);
24 44 --      DoToField('fDeviceRes', @fDeviceRes, bPoint);
24 45 --      DoToField('fEffectiveDeviceRes', @fEffectiveDeviceRes, bPoint);
24 46 --      DoToField('fViewPerPage', @fViewPerPage, bVPoint);
24 47 --      INHERITED Fields(DoToField);
24 48 -0 A  END;
24 49 --      {$ENDC}
24 50 --
24 51 --
24 52 --      {$S MANever}
24 53 --      {-----+
24 54 --      |      CalcPageStrips      |
24 55 --      +-----}
24 56 -- A  PROCEDURE TPrintHandler.CalcPageStrips (VAR pageStrips: Point);
24 57 0- A  BEGIN
24 58 --      {$IFC qDebug}
24 59 --      ProgramBreak('CalcPageStrips called for non-subclassed Printheadler');
24 60 --      {$ENDC}
24 61 -0 A  END;
24 62 --
24 63 --
24 64 --      {$S MANever}
24 65 --      {-----+
24 66 --      |      CalcViewPerPage      |
24 67 --      +-----}
24 68 -- A  PROCEDURE TPrintHandler.CalcViewPerPage(VAR amtPerPage: VPoint);
24 69 0- A  BEGIN
24 70 --      {$IFC qDebug}
24 71 --      ProgramBreak('CalcViewPerPage called for non-subclassed Printheadler');
24 72 --      {$ENDC}
24 73 -0 A  END;
24 74 --
24 75 --
24 76 --      {$S MANonRes}
24 77 --      {-----+
24 78 --      |      CheckPrinter      |
24 79 --      +-----}
24 80 -- A  PROCEDURE TPrintHandler.CheckPrinter; {See if there are changed printer-configuration parameters which
24 81 --      need to be absorbed, and if so, absorb them}
24 82 0- A  BEGIN
24 83 -0 A  END;
24 84 --
24 85 --
24 86 --      {$S BRes}
24 87 --      {-----+
24 88 --      |      DrawPrintFeedback      |
24 89 --      +-----}
24 90 -- A  PROCEDURE TPrintHandler.DrawPrintFeedback(area: Rect);
24 91 --      {Draw page-breaks, page-numbers, etc.}
24 92 0- A  BEGIN
24 93 -0 A  END;
24 94 --
24 95 --
24 96 --      {$S BRes}
24 97 --      {-----+
24 98 --      |      DrawPageBreak      |

```

```

24 99 -- +-----+
24 100 -- A PROCEDURE TPrintHandler.DrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
24 101 --                                     loc: VCoordinate; automatic: BOOLEAN);
24 102 0- A BEGIN
24 103 -0 A END;
24 104 --
24 105 --
24 106 -- {$S BRes}
24 107 -- {-----+
24 108 -- | FocusOnInterior |
24 109 -- +-----+
24 110 -- A PROCEDURE TPrintHandler.FocusOnInterior;
24 111 0- A BEGIN
24 112 -0 A END;
24 113 --
24 114 --
24 115 -- {$IFC qInspector}
24 116 -- {$S MInspector}
24 117 -- {-----+
24 118 -- | GetInspectorName |
24 119 -- +-----+
24 120 -- A PROCEDURE TPrintHandler.GetInspectorName (VAR inspectorName: Str255); OVERRIDE;
24 121 --
24 122 -- VAR
24 123 --     hexString:     String8;
24 124 --
24 125 0- A BEGIN
24 126 --     IF SELF = gNullPrintHandler THEN
24 127 --         inspectorName := 'gNullPrintHandler'
24 128 --     ELSE IF IsObject(fView) THEN
24 129 1- BEGIN
24 130 --         LIntToHex(ORD4(fView), hexString, 6);
24 131 --         InspectorName := CONCAT('for view ', hexString);
24 132 -1 END;
24 133 -0 A END;
24 134 -- {$ENDC}
24 135 --
24 136 --
24 137 -- {$S MANever}
24 138 -- {-----+
24 139 -- | LocatePageInterior |
24 140 -- +-----+
24 141 -- A PROCEDURE TPrintHandler.LocatePageInterior(pageNumber: INTEGER; VAR loc: Point);
24 142 0- A BEGIN
24 143 -0 A END;
24 144 --
24 145 --
24 146 -- {$S MANever}
24 147 -- {-----+
24 148 -- | MaxPageNumber |
24 149 -- +-----+
24 150 -- A FUNCTION TPrintHandler.MaxPageNumber: INTEGER;
24 151 0- A BEGIN
24 152 --     MaxPageNumber := 0;
24 153 -0 A END;
24 154 --
24 155 --
24 156 -- {$S MAPrint}
24 157 -- {-----+
24 158 -- | Print |
24 159 -- +-----+
24 160 -- A FUNCTION TPrintHandler.Print(itsCmdNumber: CmdNumber; VAR proceed: BOOLEAN): TCommand;
24 161 0- A BEGIN
24 162 --     proceed := TRUE;
24 163 --     Print := gNoChanges;
24 164 -0 A END;
24 165 --
24 166 --
24 167 -- {$S MANonRes}
24 168 -- {-----+
24 169 -- | PrinterChanged |
24 170 -- +-----+
24 171 -- A PROCEDURE TPrintHandler.PrinterChanged;
24 172 0- A BEGIN
24 173 -0 A END;
24 174 --
24 175 --
24 176 -- {$S MAFinder}{??? Signal error ???}
24 177 -- {-----+
24 178 -- | SetupForFinder |
24 179 -- +-----+
24 180 -- A FUNCTION TPrintHandler.SetupForFinder: BOOLEAN;
24 181 0- A BEGIN
24 182 --     {$IFC qDebug}
24 183 --         WriteLn('SetupForFinder called for a view that can't');
24 184 --     {$ENDC}
24 185 --     SetupForFinder := FALSE;
24 186 -0 A END;
24 187 --
24 188 --
24 189 -- {$S MANever}
24 190 -- {-----+
24 191 -- | RedoPageBreaks |
24 192 -- +-----+
24 193 -- A PROCEDURE TPrintHandler.RedoPageBreaks;
24 194 0- A BEGIN
24 195 -0 A END;
24 196 --
24 197 --

```

```
24 198 --      {$S BBRes}
24 199 --      {-----+
24 200 --      |   Reset
24 201 --      +-----}
24 202 -- A  PROCEDURE TPrintHandler.Reset;
24 203 0- A  BEGIN
24 204 -0 A  END;
24 205 --
24 206 --
24 207 --      {$S MANever}
24 208 --      {-----+
24 209 --      |   SetPageInterior
24 210 --      +-----}
24 211 -- A  PROCEDURE TPrintHandler.SetPageInterior(pageNumber: INTEGER);
24 212 0- A  BEGIN
24 213 -0 A  END;
24 214 --
24 215 --
24 216 --      {$S MANever}
24 217 --      {-----+
24 218 --      |   SetPageOffset
24 219 --      +-----}
24 220 -- A  PROCEDURE TPrintHandler.SetPageOffset(coord: VPoint);
24 221 0- A  BEGIN
24 222 -0 A  END;
13 3012 --
13 3013 --      END.
```

```

25 1 -- { $P }
25 2 -- { UMAUtil.p }
25 3 -- { Copyright © 1984-1988 Apple Computer, Inc. All rights reserved. }
25 4 --
25 5 -- { Conditional compile flags used by this unit:
25 6 --
25 7 --     qDebug      True includes debugging code, false ignores debugging code.
25 8 --                True also sets $N+.
25 9 --
25 10 --     qTrace      True sets $D++, causing the compiler to generate routine
25 11 --                names for the debugger, and to generate calls to %_BP
25 12 --                and %_EP at the beginning and end of each routine.
25 13 --                Furthermore, if qTrace is true, some routines are enclosed
25 14 --                with $D+ and $D++ to prevent generation of %_BP and %_EP.
25 15 --
25 16 --     qNames       Effective only if qTrace is false: If qNames is true then
25 17 --                $D+ is set, else $D- is set.
25 18 --
25 19 --     qRangeCheck  True sets $R+, false sets $R-.
25 20 --
25 21 --     qNeedsROM128K True assumes the code will always be run on 128K ROMs or
25 22 --                later and ignores code supporting 64K ROMs. False
25 23 --                includes code to support 64K ROMs.
25 24 --
25 25 -- }
25 26 --
25 27 --
25 28 -- UNIT UMAUtil;
25 29 --
25 30 -- {
25 31 --     This unit implements a set of simple type declarations and utility
25 32 --     routines.
25 33 -- }
25 34 --
25 35 -- INTERFACE
25 36 --
25 37 -- {$SETC qWWNeeded := qWriteLnWin OR qDebug OR qTrace} { Need a WriteLnWindow }
25 38 --
25 39 -- {-----+
25 40 -- | Uses |
25 41 -- +-----}
25 42 --
25 43 -- USES
25 44 --     { $LOAD MacIntf.LOAD }
25 45 --     MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
25 46 --     { $LOAD }
25 47 --     Script
25 48 -- {$IFC qDebug}
25 49 --     , UWriteLnWindow { You don't need this to use UMAUtil }
25 50 -- {$ENDC}
25 51 -- ;
25 52 --
25 53 -- {$IFC qRangeCheck}{ $R+ }{$ELSEC}{ $R- }{$ENDC}
25 54 --
25 55 -- {$IFC qTrace}{ $D+ }{$ELSEC}{ $IFC qNames}{ $D+ }{$ELSEC}{ $D- }{$ENDC}{ $ENDC }
25 56 --
25 57 -- {$IFC qDebug}{ $N+ }{$ENDC}
25 58 --
25 59 -- {-----+
25 60 -- | Global constants |
25 61 -- +-----}
25 62 --
25 63 -- CONST
25 64 --     { Character Constants }
25 65 --     chReturn      = CHR(13); { ASCII code for RETURN character }
25 66 --     chEnter        = CHR(3); { ASCII code for ENTER character }
25 67 --     chBackspace    = CHR(8); { ASCII code for BACKSPACE character }
25 68 --     chTab          = CHR(9); { ASCII code for TAB character }
25 69 --     chSpace        = CHR(32); { ASCII code for SPACE character }
25 70 --     chClear        = CHR(27); { ASCII code for CLEAR key (aka ESC) }
25 71 --     chLeft         = CHR(28); { ASCII code for left arrow }
25 72 --     chRight        = CHR(29); { ASCII code for right arrow }
25 73 --     chUp           = CHR(30); { ASCII code for up arrow }
25 74 --     chDown         = CHR(31); { ASCII code for down arrow }
25 75 --
25 76 --     kSysFontName    = ''; { GetFontNum converts this to zero. }
25 77 --     kApplFontName   = 'A'; { GetFontNum converts this to one. }
25 78 --
25 79 --     kWordAlign      = TRUE; { Constants for OffsetPtr }
25 80 --     kDontAlign      = FALSE;
25 81 --
25 82 --     kSpace8         = ' '
25 83 --                     { 12345678 }
25 84 --
25 85 --     kHexDigits      = '0123456789ABCDEF';
25 86 --
25 87 --     kNoFileRefnum   = 0; { an invalid file refnum; used to
25 88 --                          indicate an unopened file }
25 89 --
25 90 -- {$IFC qDebug}
25 91 --     { Field types for the Fields methods. These are understood by StdFieldToString. }
25 92 --     bInteger        = -1;
25 93 --     bHexInteger     = -2;
25 94 --     bLongInt        = -3;
25 95 --     bHexLongInt     = -4;
25 96 --     bString         = -5;
25 97 --     bBoolean        = -6;
25 98 --     bChar           = -7;
25 99 --     bPointer        = -8;
26 00 --     bHandle        = -9;

```



```

25 99 --      bPoint          = -10;
25 100 --     bRect          = -11;
25 101 --     bObject       = -12;
25 102 --     bByte         = -13;
25 103 --     bCmdNumber    = -14;
25 104 --     bClass        = -15;
25 105 --     bOSType        = -16;
25 106 --     bWindowPtr    = -17;
25 107 --     bControlHandle = -18;
25 108 --     bTEHandle      = -19;
25 109 --     bLowByte       = -20;
25 110 --     bHighByte      = -21;
25 111 --     bPattern       = -22;
25 112 --     bFixed         = -23;
25 113 --     bRgnHandle     = -24;
25 114 --     bRGBColor      = -25;
25 115 --     bTitle         = -26;
25 116 --     bGrafPtr       = -27;
25 117 --     bStyle         = -28;
25 118 --     bVCoordinate   = -29;
25 119 --     bVPoint        = -30;
25 120 --     bVRect         = -31;
25 121 --     bFontName      = -32;
25 122 --     bStringHandle  = -33;
25 123 --     bCntlAdornment = -34;
25 124 --     bIDType        = -36;
25 125 --     bResType       = -37;
25 126 --     { $ENDC }
25 127 --
25 128 --     {-----+
25 129 --     | Global types |
25 130 --     +-----+
25 131 --     TYPE
25 132 --         PInteger      = ^INTEGER;
25 133 --         HInteger      = ^PInteger;
25 134 --         PLongInt      = ^LONGINT;
25 135 --         HLongInt      = ^PLongInt;
25 136 --         PBoolean      = ^BOOLEAN;
25 137 --         PByte         = ^SignedByte;
25 138 --
25 139 --         String8        = STRING[8];
25 140 --         PString8      = ^String8;
25 141 --         HString8      = ^PString8;
25 142 --
25 143 --         IDType         = ResType;
25 144 --
25 145 --         adornPieces    = ( adnLineTop,      { Draw a line at the top of the extent }
25 146 --                        adnLineLeft,      { Draw a line on the left side of the extent }
25 147 --                        adnLineBottom,    { Draw a line on the bottom side of the extent }
25 148 --                        adnLineRight,    { Draw a line on the right side of the extent }
25 149 --                        adnDummy,        { Place filler... }
25 150 --                        adnOval,         { Do a FrameOval using the extent }
25 151 --                        adnRRect,        { Do a (16,16) FrameRoundRect using the extent }
25 152 --                        adnShadow );    { Draw drop shadows against framed selections }
25 153 --
25 154 --         CntlAdornment = SET OF adornPieces;
25 155 --
25 156 --         VCoordinate   = LONGINT;
25 157 --
25 158 --         VPoint        = RECORD
25 159 --             CASE INTEGER OF
25 160 --                 0: (v, h:      VCoordinate);
25 161 --                 1: (vh:        ARRAY [VHSelect] OF VCoordinate);
25 162 --             END;
25 163 --
25 164 --         VRect          = RECORD
25 165 --             CASE INTEGER OF
25 166 --                 0: (top, left, bottom, right: VCoordinate);
25 167 --                 1: (topLeft, botRight:      VPoint);
25 168 --             END;
25 169 --
25 170 --         ConfigRecord   = RECORD
25 171 --             { Values from SysEnviron (Version 1) }
25 172 --             environsVersion: INTEGER;
25 173 --             machineType:    INTEGER;
25 174 --             systemVersion:  INTEGER;
25 175 --             processor:      INTEGER;
25 176 --             hasFPU:         BOOLEAN;
25 177 --             hasColorQD:     BOOLEAN;
25 178 --             keyboardType:   INTEGER;
25 179 --             atDrvrsVersNum: INTEGER;
25 180 --             sysVRefNum:     INTEGER;
25 181 --             { Derived values }
25 182 --             hasROM128K:     BOOLEAN;
25 183 --             hasHFS:         BOOLEAN;
25 184 --             hasHierarchicalMenus: BOOLEAN;
25 185 --             hasScriptManager: BOOLEAN;
25 186 --             hasStyleTextEdit: BOOLEAN;
25 187 --             hasSoundManager: BOOLEAN;
25 188 --             hasWaitNextEvent: BOOLEAN;
25 189 --             hasSCSI:        BOOLEAN;
25 190 --             hasDesktopBus:  BOOLEAN;
25 191 --         END;
25 192 --
25 193 --     {-----+
25 194 --     | Global variables |
25 195 --     +-----+
25 196 --     VAR
25 197 --         gConfiguration: ConfigRecord; { The host machine's hardware configuration }

```

```

25 198 --      ($IFC qDebug)
25 199 --          gBoolString:      ARRAY [Boolean] OF String[5];
25 200 --                          { Used to display boolean values. }
25 201 --      ($J+)
25 202 --          GFIELDTOSTRRTN:    ProcPtr;    { Routine that converts fields to strings. }
25 203 --      ($J-)
25 204 --      ($ENDC)
25 205 --          gRGBBlack,          { RGB representation for black. }
25 206 --          gRGBWhite:          RGBColor;   { RGB representation for white. }
25 207 --
25 208 --
25 209 --      { MISCELLANEOUS UTILITIES }
25 210 --
25 211 --      ($%+)
25 212 --      FUNCTION %_GetA5: LONGINT; INLINE $2E8D:    { MOVE.L A5, (A7) }
25 213 --      FUNCTION %_GetA6: LONGINT; INLINE $2E8E:    { MOVE.L A6, (A7) }
25 214 --      FUNCTION %_GetA7: LONGINT; INLINE $2E8F:    { MOVE.L A7, (A7) }
25 215 --      ($%-)
25 216 --
25 217 --      (-----+
25 218 --      | Global Procedures |
25 219 --      +-----)
25 220 --      PROCEDURE CopyStr255 (VAR fmStr: Str255; toAddr: UNIV Ptr);
25 221 --          { Copies a string, copying ONLY what's necessary }
25 222 --
25 223 --      FUNCTION PRStr (astr: Str255): Ptr; INLINE $2E9F: { Move.L (A7)+, (A7) }
25 224 --          { For brain-damaged compiler, which can't pass the address of a string in a packed record,
25 225 --            even though it has to be byte-aligned by definition. Maybe it's not word aligned,
25 226 --            but for a BlockMove we don't care.
25 227 --            When calling CopyStr255 where the destination is the address of a Str255 in a packed
25 228 --            record, call it like this:
25 229 --            CopyStr255(fromStr, PRStr(packedStr)); }
25 230 --
25 231 --      PROCEDURE DefaultSize (VAR theSize: INTEGER);
25 232 --          { Returns 0 if theSize equals the default font size. }
25 233 --
25 234 --      PROCEDURE DefineConfiguration (VAR configuration: ConfigRecord);
25 235 --          { Fills the configuration record for the host machine. }
25 236 --
25 237 --      PROCEDURE DisposIfHandle(aHandle: UNIV Handle);
25 238 --          { Disposes the handle only if it is non-NIL. }
25 239 --
25 240 --      FUNCTION EqualBlocks(first, second: UNIV Ptr; theSize: INTEGER): BOOLEAN;
25 241 --          { Returns true if the two blocks pointed at by first and second are
25 242 --            equal over theSize bytes.}
25 243 --
25 244 --      ($IFC qDebug)
25 245 --      PROCEDURE FieldToString (theData: Ptr; fieldType: INTEGER; VAR theString: Str255);
25 246 --          { Calls the routine whose address is stored in gFieldToStrRtn. Its purpose
25 247 --            is to take some data pointed to by theData, and whose type is indicated
25 248 --            by field type, and convert that data into a string representation. By
25 249 --            default, gFieldToStrRtn points to StdFieldToString. }
25 250 --      ($ENDC)
25 251 --
25 252 --      FUNCTION GetFontNum (fontName: Str255): INTEGER;
25 253 --          { Returns the font number corresponding to the given name. If fontName is
25 254 --            equal to kSysFontName then zero is returned. If fontName is equal to
25 255 --            kApplFontName then 1 is returned. Otherwise the Toolbox routine GetFNum
25 256 --            is called to get the font number. }
25 257 --
25 258 --      FUNCTION GetHandleBits(h: Handle): SignedByte;
25 259 --          { Return the flag byte of a handle }
25 260 --
25 261 --      PROCEDURE GetIfColor(VAR aColor: RGBColor);
25 262 --          { Fetches current foreground color. Works for both old and new QuickDraw. }
25 263 --
25 264 --      PROCEDURE GetPortFontInfo (fontNum: INTEGER; VAR fontName: Str255; VAR fontSize: INTEGER);
25 265 --          { Returns the font name corresponding to a given name, in portable format.
25 266 --            i.e. if fontNum is the system font then kSysFontName is returned; if fontNum is
25 267 --            the application font then kApplFontName is return; otherwise the Toolbox routine
25 268 --            GetFontName is called to get the font name. Then, if fontNum is the system or
25 269 --            application font, a 0 is returned if the size is the default size. }
25 270 --
25 271 --      FUNCTION GetResLoad: BOOLEAN; INLINE $1EB8, $0A5E: { MOVE.B $A5E, (A7) }
25 272 --          { Returns the ResLoad flag from low memory. True indicates that resources
25 273 --            are loaded on GetResource. }
25 274 --
25 275 --      FUNCTION GetTrapType (theTrap: INTEGER): TrapType;
25 276 --          { Returns the trap's type (OSTrap or ToolTrap). }
25 277 --
25 278 --      FUNCTION IntMultiply (x, y: INTEGER): LONGINT;
25 279 --          INLINE $301F, { MOVE.W (A7)+, D0 }
25 280 --          $C1DF, { MULS.W (A7)+, D0 }
25 281 --          $2E80: { MOVE.L D0, (A0) }
25 282 --          { Multiplies two integers and returns a longint result. Note that the
25 283 --            Pascal compiler will generate a 16-bit result if you say n := x * y,
25 284 --            truncating the upper 16-bits. Furthermore, if either x or y is a
25 285 --            longint then the compiler expands both operands to 32-bits and calls
25 286 --            a 32-bit multiply subroutine. IntMultiply avoids these problems. }
25 287 --
25 288 --      * FUNCTION LengthRect(r: Rect; vhs: VHSelect): INTEGER;
25 289 --          { If vhs is v, returns the length of r; else returns the
25 290 --            width of r. }
25 291 --
25 292 --      PROCEDURE LIntToHex (decNumber: LONGINT; VAR hexNumber: String8;
25 293 --          noOfDigits: INTEGER);
25 294 --          { LIntToHex converts decNumber to a hexadecimal string of
25 295 --            noOfDigits length. }
25 296 --

```

```

25 297 -- FUNCTION Max(a, b: LONGINT): LONGINT;
25 298 --     { Returns the maximum of a and b. }
25 299 --
25 300 -- FUNCTION Min(a, b: LONGINT): LONGINT;
25 301 --     { Returns the minimum of a and b. }
25 302 --
25 303 -- FUNCTION NewSysPtr (noOfBytes: LONGINT): Ptr;
25 304 --     { Creates a new point of noOfBytes length in the system heap.}
25 305 --
25 306 -- PROCEDURE NumberToHex (theNumber: LONGINT; VAR hexString: Str255;
25 307 --     hexDigits: INTEGER);
25 308 --     { Converts theNumber to a hex string preceeded with '$'.}
25 309 --
25 310 -- FUNCTION PinOnRect(theRect: Rect; thePt: Point): LONGINT;
25 311 --     { Like PinRect except that if thePt.h (or v) is >= theRect.right (bottom)
25 312 --     it returns theRect.right (bottom). PinRect returns 1 less than that
25 313 --     except if thePt is exactly on the edge of theRect.}
25 314 --
25 315 -- PROCEDURE PointerToHex (theNumber: LONGINT; VAR hexString: Str255;
25 316 --     hexDigits: INTEGER);
25 317 --     { If theNumber is zero, then hexString is set to 'Nil'. Else theNumber
25 318 --     is converted to a hex string preceeded with '$'.}
25 319 --
25 320 -- FUNCTION RectsNest(outer, inner: Rect): BOOLEAN;
25 321 --     { Determine if inner nests within outer. }
25 322 --
25 323 -- FUNCTION RoundUp(aNumber: LONGINT; aModulus: INTEGER): LONGINT;
25 324 --     { Rounds aNumber up so that it is evenly divisible by aModulus. }
25 325 --
25 326 -- PROCEDURE SetKeyScript (newKeyScript: INTEGER);
25 327 --     { If newKeyScript is different from the current key script, then the
25 328 --     key script is set to newKeyScript. (The reason we don't want to
25 329 --     set the key script to the same thing is that the Script Mgr.
25 330 --     does a FlushEvents when it sets the key script.) }
25 331 --
25 332 -- PROCEDURE SetHandleBits(h: Handle; theBits: SignedByte);
25 333 --     { Sets the flag byte of a handle }
25 334 --
25 335 -- PROCEDURE SetIfColor(aColor: RGBColor);
25 336 --     { Sets foreground color. Works for both old and new QuickDraw. }
25 337 --
25 338 -- PROCEDURE SetPortTextStyle (theTextStyle: TextStyle);
25 339 --     { Sets the font style of the current port to the characteristics of
25 340 --     theTextStyle. }
25 341 --
25 342 -- PROCEDURE SetRGBColor (VAR RGB: RGBColor; red, green, blue: INTEGER);
25 343 --     { Sets RGB to the given colors. }
25 344 --
25 345 -- PROCEDURE SetTextStyle(VAR theTextStyle: TextStyle; theFont: INTEGER;
25 346 --     theStyle: Style; theSize: INTEGER; theColor: RGBColor);
25 347 --     { Sets theTextStyle to the given characteristics }
25 348 --
25 349 -- {$IFC qDebug}
25 350 -- PROCEDURE StdFieldToString (theData: Ptr; fieldType: INTEGER; VAR theString: Str255);
25 351 --     { This routine converts all field types defined by the constants in this unit. }
25 352 -- {$ENDC}
25 353 --
25 354 -- {$IFC qDebug}
25 355 -- PROCEDURE TextStyleFields (aTitle: Str255; VAR aStyle: TextStyle;
25 356 --     PROCEDURE DoToField (fieldName: Str255;
25 357 --         fieldAddr: Ptr;
25 358 --         fieldType: INTEGER));
25 359 --     { Used to inspect the fields of a TextStyle record. The
25 360 --     record is passed as a var parameter because we must be passed
25 361 --     the address of the original record being referenced. Passing
25 362 --     by value would simply make a copy of the record, which doesn't
25 363 --     work for the inspector. }
25 364 -- {$ENDC}
25 365 --
25 366 -- FUNCTION UpChar (ch: CHAR): CHAR;
25 367 --     { Returns ch converted to upper case }
25 368 --
25 369 -- PROCEDURE UpString8 (VAR s: String8);
25 370 --     { Converts s to all upper case. }
25 371 --
25 372 -- PROCEDURE UseROMMap(resLoad: BOOLEAN);
25 373 --     { Call this before any Resource Manager call for which you might like to use
25 374 --     ROM resources. Remember, if you pass FALSE to this then you should
25 375 --     call SetResLoad(TRUE) afterwards. }
25 376 --
25 377 --
25 378 -- { F I L E   U T I L I T I E S }
25 379 --
25 380 -- FUNCTION CloseFile (dataRefnum, rsrcRefnum: INTEGER): OSErr;
25 381 --     { Closes the data and resource forks of a file. If dataRefnum =
25 382 --     kNoFileRefnum then the data fork is not closed. Likewise, if
25 383 --     rsrcRefnum = kNoFileRefnum then the resource fork is not closed.
25 384 --     Returns noErr if successful, else an O.S. error. }
25 385 --
25 386 -- FUNCTION DeleteFile(namePtr: StringPtr; volRefnum: INTEGER): OSErr;
25 387 --     { Deletes the specified file; uses FillInDirID to bypass the
25 388 --     Poor Man's Search Path. }
25 389 --
25 390 -- FUNCTION FileModDate(name: Str255; volRefnum: INTEGER): LONGINT;
25 391 --     { Returns file modification date or 0 if an I/O error occurs }
25 392 --
25 393 -- FUNCTION FillInDirID(pb: HParamBlkPtr): OSErr;
25 394 --     { Based on the ioVRefnum field of pb, looks up the dirID of the
25 395 --     working directory and fills it into the ioDirID field. If HFS

```

```

25 396 --      is not installed, it sets the ioDirID to 0.
25 397 --
25 398 --      This is used to inhibit the PMSP. After setting up your HParamBlockRec,
25 399 --      call this to fill in the dirID and then make the H form of the
25 400 --      call (eg., PBHGetFInfo rather than PBGetFInfo). )
25 401 --
25 402 --      FUNCTION GetDirID(VAR vRefnum: INTEGER; VAR dirID: LONGINT): OSErr;
25 403 --      { Returns the dirID corresponding to a given wdRefnum; changes vRefnum
25 404 --      to be a real volume refnum. }
25 405 --
25 406 --      FUNCTION GetFileInfo(name: Str255; volRefnum: INTEGER; VAR info: HParamBlockRec): OSErr;
25 407 --      { Makes a PBHGetFInfo call, and returns result in info. Bypasses PMSP. }
25 408 --
25 409 --      FUNCTION NumBlocks(numBytes: LONGINT; blkSize: LONGINT): LONGINT;
25 410 --      { Returns the number of blocks required to store numBytes, given that
25 411 --      the block size is blkSize. }
25 412 --
25 413 --      FUNCTION OpenFile(name: Str255; volRefnum: INTEGER;
25 414 --      openData, openRsrc: BOOLEAN; dataPerm, rsrcPerm: INTEGER;
25 415 --      VAR dataRefnum, rsrcRefnum: INTEGER): OSErr;
25 416 --      { Open the specified forks of the file using the specified permissions.
25 417 --      Returns kNoFileRefnum for refnums that are not (or cannot) be opened.
25 418 --      On 64K ROM machines, rsrcPerm is ignored. Returns an O/S error if
25 419 --      the data fork does not exist; returns a rsrcRefnum of kNoFileRefnum
25 420 --      if the resource fork does not exist. Returns an O/S error for all
25 421 --      other errors. If no error occurs, returns NoErr. }
25 422 --
25 423 --
25 424 --      ( D E B U G G I N G   U T I L I T I E S )
25 425 --
25 426 --      ($IFC qDebug)
25 427 --      FUNCTION CanReadLn: BOOLEAN;
25 428 --      { returns TRUE if WriteLnWindow is active and the display is not suppressed }
25 429 --
25 430 --      PROCEDURE GetMethodName (ppc: LONGINT; VAR s: Str255);
25 431 --      { GetMethodName returns the name of the method (or procedure) in
25 432 --      which ppc points. }
25 433 --
25 434 --      PROCEDURE GetProcName(pPC: LONGINT; VAR className, procName: String8);
25 435 --      { GetProcName returns the name of the procedure or function in
25 436 --      which ppc points. If it is in a method, then it return's
25 437 --      the name of the method's class in className. }
25 438 --
25 439 --      PROCEDURE MacsBug; INLINE $A9FF;
25 440 --      { Causes a program break to MacsBug. }
25 441 --
25 442 --      PROCEDURE MacsBugStr(s: Str255); INLINE $ABFF;
25 443 --      { Causes a program break to MacsBug--MacsBug displays the message s. }
25 444 --
25 445 --      FUNCTION ReadInteger (prompt: Str255): INTEGER;
25 446 --      { Displays prompt and reads an integer from the debug window. }
25 447 --
25 448 --      FUNCTION ReadYesNo (prompt: Str255): BOOLEAN;
25 449 --      { Displays prompt and accepts a 'y' or 'n' as input from the debug
25 450 --      window. }
25 451 --
25 452 --      PROCEDURE WritePt(pt: Point);
25 453 --      { Writes the point in the debug window. }
25 454 --
25 455 --      PROCEDURE WritePtr(val: UNIV LONGINT);
25 456 --      { Writes the pointer, in hex, in the debug window. }
25 457 --
25 458 --      PROCEDURE WriteRect(r: Rect);
25 459 --      { Writes the rectangle in the debug window. }
25 460 --
25 461 --      PROCEDURE WriteBoolean(b: BOOLEAN);
25 462 --      { Writes the boolean in the debug window. }
25 463 --
25 464 --      PROCEDURE WriteVPt (pt: VPoint);
25 465 --      { Writes the VPoint in the debug window. }
25 466 --
25 467 --      PROCEDURE WriteVRect (r: VRect);
25 468 --      { Writes the VRect in the debug window. }
25 469 --
25 470 --      PROCEDURE WrLblPt(aLabel: Str255; pt: Point);
25 471 --      { Writes the label, ' - ', the point, to the debug window. }
25 472 --
25 473 --      PROCEDURE WrLblPtr(aLabel: Str255; val: UNIV LONGINT);
25 474 --      { Writes the label, ' - ', the pointer in hex, to the debug window. }
25 475 --
25 476 --      PROCEDURE WrLblRect(aLabel: Str255; r: Rect);
25 477 --      { Writes the label, ' - ', the rectangle, to the debug window. }
25 478 --
25 479 --      PROCEDURE WrLblBoolean(aLabel: Str255; b: BOOLEAN);
25 480 --      { Writes the label, ' - ', the boolean, to the debug window. }
25 481 --
25 482 --      PROCEDURE WrLblVPt(aLabel: Str255; pt: VPoint);
25 483 --      { Writes the label, ' - ', the VPoint in the debug window. }
25 484 --
25 485 --      PROCEDURE WrLblVRect (aLabel: Str255; r: VRect);
25 486 --      { Writes the lable, ' - ', the VRect in the debug window. }
25 487 --
25 488 --      PROCEDURE WriteSig(theID: IDType);
25 489 --
25 490 --      PROCEDURE WrLblSig(theLabel: Str255; theID: IDType);
25 491 --
25 492 --      PROCEDURE WriteHexInt(theInt: INTEGER);
25 493 --
25 494 --      PROCEDURE WrLblHexInt(theLabel: Str255; theInt: INTEGER);

```

```
25 495 --  
25 496 -- PROCEDURE WriteHexLongint(theLongint: LONGINT);  
25 497 --  
25 498 -- PROCEDURE WrLblHexLongint(theLabel: Str255; theLongint: LONGINT);  
25 499 -- {SENDC qDebug}  
25 500 --  
25 501 --  
25 502 -- IMPLEMENTATION  
25 503 --  
25 504 -- {$I UMAUtil.incl.p}
```

```

26 1 --      {$P}
26 2 --      {UMAUtl.incl.p}
26 3 --      {Copyright © 1984-1988 Apple Computer, Inc. All rights reserved.}
26 4 --
26 5 --      {$IFC qDebug}
26 6 --      {$IFC qTrace}{$D+}{$ENDC}
26 7 --      {$S MADEbug}
26 8 --      {-----+
26 9 --      | CanReadLn |
26 10 --      +-----}
26 11 -- A FUNCTION CanReadLn: BOOLEAN;
26 12 --
26 13 0- A BEGIN
26 14 --      CanReadLn := FALSE;
26 15 --      IF gDebugWindowPtr <> NIL THEN
26 16 --          CanReadLn := gWrToWindow;
26 17 -0 A END {CanReadLn};
26 18 --      {$IFC qTrace}{$D++}{$ENDC}
26 19 --      {$ENDC qDebug}
26 20 --
26 21 --      {$S MAFile}
26 22 --      {-----+
26 23 --      | CloseFile |
26 24 --      +-----}
26 25 -- A FUNCTION CloseFile (dataRefnum, rsrcRefnum: INTEGER): OSErr;
26 26 --
26 27 --      VAR
26 28 --          err: OSErr;
26 29 --
26 30 0- A BEGIN
26 31 --          err := noErr;
26 32 --
26 33 --          IF dataRefnum <> kNoFileRefnum THEN
26 34 --              err := FSClose(dataRefnum);
26 35 --
26 36 --          IF rsrcRefnum <> kNoFileRefnum THEN
26 37 1-              BEGIN
26 38 --                  CloseResFile(rsrcRefnum);
26 39 --                  IF err = noErr THEN
26 40 --                      err := ResError;
26 41 -1                      END;
26 42 --
26 43 --                  CloseFile := err;
26 44 -0 A END {CloseFile};
26 45 --
26 46 --      {$S MAMain}
26 47 --      {-----+
26 48 --      | CopyStr255 |
26 49 --      +-----}
26 50 -- A PROCEDURE CopyStr255 (VAR fmStr: Str255; toAddr: UNIV Ptr);
26 51 0- A BEGIN
26 52 --      BlockMove(@fmStr, toAddr, LENGTH(fmStr)+1);
26 53 -0 A END;
26 54 --
26 55 --      {$S MAMain}
26 56 --      {-----+
26 57 --      | DefaultSize |
26 58 --      +-----}
26 59 -- A PROCEDURE DefaultSize (VAR theSize: INTEGER);
26 60 --      CONST
26 61 --          SysFontSize = $BA8;
26 62 0- A BEGIN
26 63 --          IF gConfiguration.hasROM128K &
26 64 --              (gConfiguration.hasScriptManager & (theSize = GetDefFontSize)) |
26 65 --              (theSize = PInteger(SysFontSize)^) THEN
26 66 --              theSize := 0;
26 67 -0 A END;
26 68 --
26 69 --      {$IFC qTrace}{$D+}{$ENDC}
26 70 --      {$S MAInit}
26 71 --      {-----+
26 72 --      | DefineConfiguration |
26 73 --      +-----}
26 74 -- A PROCEDURE DefineConfiguration (VAR configuration: ConfigRecord);
26 75 --
26 76 --      CONST
26 77 --          {Low Memory Globals}
26 78 --          ROM85 = $28E;
26 79 --          HwCfgFlags = $B22;
26 80 --          FSFCBLen = $3F6;
26 81 --
26 82 --          {Masks for the HwCfgFlags}
26 83 --          mSCSIPort = $8000;
26 84 --          mDesktopBus = $0400;
26 85 --
26 86 --          {Traps used to determine the presence of features}
26 87 --          UnimplementedTrap = $9F;
26 88 --          tScriptUtil = $A8B5; { determines presence of Script Manager }
26 89 --          tPopupMenuSelect = $A80B; { determines whether ROM has hier. menu support }
26 90 --          tSndDoCommand = $A803; { determines presence of Sound Manager }
26 91 --          tWaitNextEvent = $A860; { determines presence of WaitNextEvent }
26 92 --
26 93 --      VAR
26 94 --          kludge: ^SysEnvRec;
26 95 --          result: OSErr;
26 96 --
26 97 --      {-----+
26 98 --      | TrapExists |

```

```

26 99 -- +-----}
26 100 -- B FUNCTION TrapExists (theTrap: INTEGER): BOOLEAN;
26 101 --
26 102 0- B BEGIN
26 103 --     TrapExists := GetTrapAddress(UnimplementedTrap) <>
26 104 --             NGetTrapAddress(theTrap, GetTrapType(theTrap));
26 105 0- B END {TrapExists};
26 106 --
26 107 0- A BEGIN
26 108 --     kludge := @configuration;
26 109 --     result := SysEnviron(1, kludge);      {Version 1 shouldn't fail}
26 110 --
26 111 --     WITH configuration DO
26 112 1- BEGIN
26 113 --         hasDesktopBus := BAND(pInteger(HwCfgFlags)^, mDesktopBus) > 0;
26 114 --         hasSCSI := BAND(pInteger(HwCfgFlags)^, mSCSIPort) > 0;
26 115 --         hasROM128K := machineType > envMac;
26 116 --         IF hasROM128K THEN
26 117 --             hasHFS := TRUE
26 118 --         ELSE
26 119 --             hasHFS := PInteger(FSFCBLen)^ > 0;
26 120 --             hasHierarchicalMenus := hasROM128K AND TrapExists(tPopupMenuSelect);
26 121 --             hasScriptManager := hasROM128K AND TrapExists(tScriptUtil);
26 122 --             hasStyleTextEdit := systemVersion >= $600;
26 123 --             hasSoundManager := hasROM128K AND TrapExists(tSndDoCommand);
26 124 --             hasWaitNextEvent := hasROM128K AND TrapExists(tWaitNextEvent);
26 125 -1 END {with};
26 126 0- A END {DefineConfiguration};
26 127 --     {$IFC qTrace}{$D+}{$ENDC}
26 128 --
26 129 --     {$S MAFFile}
26 130 --     {-----+
26 131 --     | DeleteFile |
26 132 --     +-----}
26 133 -- A FUNCTION DeleteFile(namePtr: StringPtr; volRefnum: INTEGER): OSErr;
26 134 --     VAR hPB: HParamBlockRec;
26 135 --     err: OSErr;
26 136 0- A BEGIN
26 137 --     WITH hPB DO
26 138 1- BEGIN
26 139 --         ioNamePtr := namePtr;
26 140 --         ioVRefnum := volRefnum;
26 141 --         ioFVersNum := 0;
26 142 -1 END;
26 143 --
26 144 --     err := FillInDirID(@hPB);      {to avoid PMSF}
26 145 --
26 146 --     IF err = noErr THEN
26 147 --         err := PBHDelete(@hPB, FALSE);
26 148 --
26 149 --         DeleteFile := err;
26 150 0- A END {DeleteFile};
26 151 --
26 152 --     {$S MAMain}
26 153 --     {-----+
26 154 --     | DisposIfHandle |
26 155 --     +-----}
26 156 -- A PROCEDURE DisposIfHandle (aHandle: UNIV Handle);
26 157 --
26 158 0- A BEGIN
26 159 --     IF aHandle <> NIL THEN
26 160 --         DisposHandle(aHandle);
26 161 0- A END {DisposIfHandle};
26 162 --
26 163 --     {-----+
26 164 --     | EqualBlocks |
26 165 --     +-----}
26 166 -- A FUNCTION EqualBlocks(first, second: UNIV Ptr; theSize: INTEGER): BOOLEAN; EXTERNAL;
26 167 --
26 168 --     {$S MAFFile}
26 169 --     {-----+
26 170 --     | FileModDate |
26 171 --     +-----}
26 172 -- A FUNCTION FileModDate(name: Str255; volRefnum: INTEGER): LONGINT;
26 173 --     VAR pb: HParamBlockRec;
26 174 --
26 175 0- A BEGIN
26 176 --     IF GetFileInfo(name, volRefnum, pb) = noErr THEN
26 177 --         FileModDate := pb.ioFlMdDat
26 178 --     ELSE
26 179 --         FileModDate := 0;
26 180 0- A END {FileModDate};
26 181 --
26 182 --     {$IFC qDebug}
26 183 --     {-----+
26 184 --     | FieldToString |
26 185 --     +-----}
26 186 -- A PROCEDURE FieldToString (theData: Ptr; fieldType: INTEGER; VAR theString: Str255); EXTERNAL;
26 187 --     {$ENDC}
26 188 --
26 189 --
26 190 --     {$S MAFFile}
26 191 --     {-----+
26 192 --     | FillInDirID |
26 193 --     +-----}
26 194 -- A FUNCTION FillInDirID(pb: HParamBlkPtr): OSErr;
26 195 --
26 196 0- A BEGIN
26 197 --     FillInDirID := GetDirID(pb^.ioVRefnum, pb^.ioDirID);

```

```

26 198 -0 A  END (FillInDirID);
26 199 --
26 200 --  {$S MAFile}
26 201 --  {-----+
26 202 --  |  GetDirID
26 203 --  +-----}
26 204 -- A  FUNCTION  GetDirID (VAR vRefnum: INTEGER; VAR dirID: LONGINT): OSErr;
26 205 --      VAR pb: WDPBRec;
26 206 --
26 207 0- A  BEGIN
26 208 --      IF gConfiguration.hasHFS THEN
26 209 1-          BEGIN
26 210 --              WITH pb DO
26 211 2-                  BEGIN
26 212 --                      ioNamePtr := NIL;
26 213 --                      ioVRefnum := vRefnum;
26 214 --                      ioWDIndex := 0;
26 215 --                      ioWDProcID := 0;
26 216 --                      ioWDVRefnum := vRefnum;
26 217 -2                      END;
26 218 --                      GetDirID := PBGetWDInfo(@pb, FALSE);
26 219 --                      vRefnum := pb.ioWDVRefnum;
26 220 --                      dirID := pb.ioWDDirID;
26 221 -1                      END
26 222 --              ELSE
26 223 1-                  BEGIN
26 224 --                      dirID := 0;
26 225 --                      GetDirID := noErr;
26 226 -1                      END;
26 227 -0 A  END (GetDirID);
26 228 --
26 229 --  {$S MAFile}
26 230 --  {-----+
26 231 --  |  GetFileInfo
26 232 --  +-----}
26 233 -- A  FUNCTION  GetFileInfo (name: Str255; volRefnum: INTEGER; VAR info: HParamBlockRec): OSErr;
26 234 --      VAR err: OSErr;
26 235 0- A  BEGIN
26 236 --      WITH info DO
26 237 1-          BEGIN
26 238 --              ioNamePtr := @name;
26 239 --              ioVRefnum := volRefnum;
26 240 --              ioFVersNum := 0;
26 241 --              ioFDirIndex := 0;
26 242 -1              END;
26 243 --              err := FillInDirID(@info);
26 244 --              IF err = noErr THEN
26 245 --                  err := PBHGetFInfo(@info, FALSE);
26 246 --              GetFileInfo := err;
26 247 -0 A  END (GetFileInfo);
26 248 --
26 249 --  {$S MAMain}
26 250 --  {-----+
26 251 --  |  GetFontNum
26 252 --  +-----}
26 253 -- A  FUNCTION  GetFontNum (fontName: Str255): INTEGER;
26 254 --
26 255 --      VAR
26 256 --          fontNum: INTEGER;
26 257 --
26 258 0- A  BEGIN
26 259 --      UprString(fontName, FALSE);
26 260 --      IF fontName = kSysFontName THEN
26 261 --          fontNum := systemFont
26 262 --      ELSE IF fontName = kApplFontName THEN
26 263 --          fontNum := applFont
26 264 --      ELSE
26 265 --          GetFNum(fontName, fontNum);
26 266 --          GetFontNum := fontNum;
26 267 -0 A  END;
26 268 --
26 269 --  {$S Main} {Must be in Main segment and cannot call to any other segment.}
26 270 --  {$IFC qTrace}{$D+}{$ENDC}
26 271 --  {-----+
26 272 --  |  GetHandleBits
26 273 --  +-----}
26 274 -- A  FUNCTION  GetHandleBits (h: Handle): SignedByte;
26 275 --
26 276 0- A  BEGIN
26 277 --      {$IFC NOT qNeedsROM128K}
26 278 --          IF NOT gConfiguration.hasROM128K THEN
26 279 --              GetHandleBits := PByte(h)^
26 280 --          ELSE
26 281 --              {$ENDC}
26 282 --              GetHandleBits := HGetState(h);
26 283 -0 A  END;
26 284 --      {$IFC qTrace}{$D++}{$ENDC}
26 285 --
26 286 --  {$S MAMain}
26 287 --  {-----+  Get present foreground color
26 288 --  |  GetIfColor  Works with both old and new QuickDraw
26 289 --  +-----}
26 290 -- A  PROCEDURE  GetIfColor (VAR aColor: RGBColor);
26 291 --      CONST
26 292 --          BlackBit   = 5;
26 293 --          YellowBit   = 6;
26 294 --          MagentaBit  = 7;
26 295 --          CyanBit     = 8;
26 296 --      VAR

```



```

26 297 --      oldColor:  LONGINT;
26 298 0- A  BEGIN
26 299 --      IF gConfiguration.hasColorQD THEN
26 300 --          GetForeColor(aColor)
26 301 --      ELSE
26 302 1-          BEGIN
26 303 --              (*          xxxxxxxx C.MY B rgb w b = RGB          *)
26 304 --              blackColor  = 33 = 0000000 0.00 1 000 0 1 = 000
26 305 --              whiteColor  = 30 = 0000000 0.00 0 111 1 0 = 111
26 306 --              redColor    = 205 = 0000000 0.11 0 011 0 1 = 100
26 307 --              greenColor  = 341 = 0000000 1.01 0 101 0 1 = 010
26 308 --              blueColor   = 409 = 0000000 1.10 0 110 0 1 = 001
26 309 --              cyanColor   = 273 = 0000000 1.00 0 100 0 1 = 011
26 310 --              magentaColor = 137 = 0000000 0.10 0 010 0 1 = 101
26 311 --              yellowColor  = 69 = 0000000 0.01 0 001 0 1 = 110
26 312 --          *)
26 313 --          oldColor := thePort^.fgColor;          { Fetch old color          }
26 314 --          aColor := gRGBBlack;                  { Prime returned color to black    }
26 315 --          IF BTST(oldColor, BlackBit) THEN      { If color isn't black, force CMY = 111 }
26 316 --              oldColor := BOR(oldColor, $1C0);
26 317 --          IF NOT BTST(oldColor, CyanBit) THEN   { Absence of cyan = presence of red  }
26 318 --              aColor.red := $FFFF;
26 319 --          IF NOT BTST(oldColor, MagentaBit) THEN { Absence of magenta = presence of green }
26 320 --              aColor.green := $FFFF;
26 321 --          IF NOT BTST(oldColor, YellowBit) THEN { Absence of yellow = presence of blue }
26 322 --              aColor.blue := $FFFF;
26 323 1-      END;
26 324 0- A  END;
26 325 --
26 326 --      { $IFC qDebug }
26 327 --      { $S MAdDebug }
26 328 --      {-----+
26 329 --      |  GetMethodName
26 330 --      +-----}
26 331 0- A  PROCEDURE GetMethodName (ppc: LONGINT; VAR s: Str255);
26 332 --
26 333 --      VAR
26 334 --          c:      String8;
26 335 --          m:      String8;
26 336 --          x:      INTEGER;
26 337 --
26 338 0- A  BEGIN
26 339 --      GetProcName(ppc, c, m);
26 340 --
26 341 --      IF c = kSpace8 THEN
26 342 --          s := m
26 343 --      ELSE
26 344 --          s := CONCAT(c, '.', m);
26 345 1-      REPEAT
26 346 --          x := POS(' ', s);
26 347 --          IF x > 0 THEN
26 348 --              Delete(s, x, 1);
26 349 1-      UNTIL x = 0;
26 350 0- A  END (GetMethodName);
26 351 --      { $ENDC qDebug }
26 352 --
26 353 --      { $S MAMain }
26 354 --      {-----+
26 355 --      |  GetPortFontInfo
26 356 --      +-----}
26 357 0- A  PROCEDURE GetPortFontInfo (fontNum: INTEGER; VAR fontName: Str255; VAR fontSize: INTEGER);
26 358 --      CONST
26 359 --          ApFontID = $984;
26 360 --          SysFontFam = $BA6;
26 361 --
26 362 0- A  BEGIN
26 363 --      IF (fontNum = systemFont) |
26 364 --          (gConfiguration.hasROM128K &
26 365 --          ((gConfiguration.hasScriptManager & (fontNum = GetSysFont)) |
26 366 --          (fontNum = PInteger(SysFontFam)))) THEN
26 367 1-          BEGIN
26 368 --              fontName := kSysFontName;
26 369 --              DefaultSize(fontSize);
26 370 1-      END
26 371 --
26 372 --      ELSE IF (fontNum = applFont) |
26 373 --          (gConfiguration.hasScriptManager & (fontNum = GetAppFont)) |
26 374 --          (fontNum = PInteger(ApFontID)) THEN
26 375 1-          BEGIN
26 376 --              fontName := kApplFontName;
26 377 --              DefaultSize(fontSize);
26 378 1-      END
26 379 --
26 380 --      ELSE
26 381 --          GetFontName(fontNum, fontName);
26 382 0- A  END;
26 383 --
26 384 --      { $IFC qDebug }
26 385 --      { $MAdDebug }
26 386 --      {-----+
26 387 --      |  GetProcName
26 388 --      +-----}
26 389 0- A  PROCEDURE GetProcName(ppc: LONGINT; VAR className, procName: String8);
26 390 --
26 391 --      VAR
26 392 --          pname:      PByte;
26 393 --          pc:          PInteger;
26 394 --          bothClassAndProc:  BOOLEAN;
26 395 --

```

```

26 396 --      {-----+
26 397 --      |   CopyName   |
26 398 --      +-----+
26 399 -- B      PROCEDURE CopyName(VAR anyName: String8);
26 400 --
26 401 --      VAR
26 402 --          j: INTEGER;
26 403 --
26 404 -- B      BEGIN
26 405 --          anyName := kSpace8;
26 406 --          FOR j := 1 TO 8 DO
26 407 --              BEGIN
26 408 --                  anyName[j] := CHR(BAND(pname*, 127));
26 409 --                  pname := PByte(ORD(pname) + 1);
26 410 --              END;
26 411 -- B      END;
26 412 --
26 413 -- A      BEGIN
26 414 --          className := kSpace8;
26 415 --          procName := kSpace8;
26 416 --
26 417 --          pc := HInteger(pPC)*;
26 418 --
26 419 --          IF (ORD(pc) <> 0) AND NOT ODD(ORD(pc)) THEN
26 420 --              BEGIN
26 421 --                  {We add the -1 to the following tests so that the things we are
26 422 --                      searching for don't appear in the body of the procedure.}
26 423 --
26 424 --                  {Search for UNLK A6.}
26 425 --                  WHILE pc* - 1 <> $4E5E - 1 DO
26 426 --                      pc := PInteger(ORD(pc) + 2);
26 427 --
26 428 --                  {Search for RTS or JMP (A0) or RTD #$00nn} {Thanks to Mike Bentley}
26 429 --                  WHILE (pc* - 1 <> $4E75 - 1) &
26 430 --                      (pc* - 1 <> $4ED0 - 1) &
26 431 --                      (pc* - 1 <> $4E74 - 1) DO
26 432 --                      pc := PInteger(ORD(pc) + 2);
26 433 --
26 434 --                  IF pc* - 1 = $4E74 - 1 THEN {If it was RTD, skip past displacement}
26 435 --                      pc := PInteger(ORD4(pc) + 2);
26 436 --                      pname := PByte(ORD(pc) + 2); {pname read and changed by CopyName}
26 437 --                      IF pname* < 0 THEN
26 438 --                          BEGIN
26 439 --                              CopyName(procName);
26 440 --                              IF PByte(ORD(pc) + 3)* < 0 THEN
26 441 --                                  CopyName(String8(className))
26 442 --                              ELSE
26 443 --                                  className := kSpace8;
26 444 --                              END;
26 445 --                          END;
26 446 --                      END;
26 447 -- A      END {GetProcName};
26 448 --      {SEND C qDebug}
26 449 --
26 450 --      {$S Main}
26 451 --      {-----+
26 452 --      |   GetTrapType   |
26 453 --      +-----+
26 454 -- A      FUNCTION GetTrapType (theTrap: INTEGER): TrapType;
26 455 --
26 456 -- A      BEGIN
26 457 --          { OS traps start with A0, Tool with A8 or AA. }
26 458 --          IF BAND(theTrap, $0F00) = 0 THEN
26 459 --              GetTrapType := OSTrap
26 460 --          ELSE
26 461 --              GetTrapType := ToolTrap;
26 462 -- A      END {GetTrapType};
26 463 --
26 464 --      {$S MAMain}
26 465 --      {-----+
26 466 --      |   LengthRect   |
26 467 --      +-----+
26 468 -- A      FUNCTION LengthRect (r: Rect; vhs: VHSelect): INTEGER;
26 469 --
26 470 -- A      BEGIN
26 471 --          LengthRect := r.botRight.vh[vhs] - r.topLeft.vh[vhs];
26 472 -- A      END {LengthRect};
26 473 --
26 474 --      {IFC qTrace}{SD+}{SEND C}
26 475 --      {$S MADebug}
26 476 --      {-----+
26 477 --      |   LIntToHex   |
26 478 --      +-----+
26 479 -- A      PROCEDURE LIntToHex (decNumber: LONGINT; VAR hexNumber: String8;
26 480 --                          noOfDigits: INTEGER);
26 481 --      VAR
26 482 --          i:          INTEGER;
26 483 --
26 484 -- A      BEGIN
26 485 --          noOfDigits := Min(noOfDigits, 8);
26 486 --          hexNumber[0] := CHR(noOfDigits);
26 487 --          FOR i := noOfDigits DOWNT0 1 DO
26 488 --              BEGIN
26 489 --                  hexNumber[i] := kHexDigits[BAND(decNumber, 15) + 1];
26 490 --                  decNumber := BSR(decNumber, 4);
26 491 --              END;
26 492 -- A      END {LIntToHex};
26 493 --      {IFC qTrace}{SD++}{SEND C}
26 494 --

```

```

26 495 --      {$IFC qTrace}{$D+}{$ENDC}
26 496 --      {$S MAMain}
26 497 --      {-----+
26 498 --      |      Max
26 499 --      +-----+}
26 500 -- A  FUNCTION Max(a, b: LONGINT): LONGINT;
26 501 0- A  BEGIN
26 502 --      IF a < b THEN
26 503 --          Max := b
26 504 --      ELSE
26 505 --          Max := a;
26 506 0- A  END {Max};
26 507 --      {$IFC qTrace}{$D+}{$ENDC}
26 508 --
26 509 --      {$IFC qTrace}{$D+}{$ENDC}
26 510 --      {$S MAMain}
26 511 --      {-----+
26 512 --      |      Min
26 513 --      +-----+}
26 514 -- A  FUNCTION Min(a, b: LONGINT): LONGINT;
26 515 0- A  BEGIN
26 516 --      IF a < b THEN
26 517 --          Min := a
26 518 --      ELSE
26 519 --          Min := b;
26 520 0- A  END {Min};
26 521 --      {$IFC qTrace}{$D+}{$ENDC}
26 522 --
26 523 --      {-----+
26 524 --      |      NumberToHex
26 525 --      +-----+}
26 526 -- A  FUNCTION NewSysPtr (noOfBytes: LONGINT): Ptr: EXTERNAL;
26 527 --
26 528 --      {$IFC qTrace}{$D+}{$ENDC}
26 529 --      {$S MADEBUG}
26 530 --      {-----+
26 531 --      |      NumberToHex
26 532 --      +-----+}
26 533 -- A  PROCEDURE NumberToHex (theNumber: LONGINT; VAR hexString: Str255; hexDigits: INTEGER);
26 534 --
26 535 --      VAR
26 536 --          tempString:      String8;
26 537 --
26 538 0- A  BEGIN
26 539 --          LIntToHex(theNumber, tempString, hexDigits);
26 540 --          hexString := CONCAT('$', tempString);
26 541 0- A  END {NumberToHex};
26 542 --      {$IFC qTrace}{$D+}{$ENDC}
26 543 --
26 544 --      {$IFC qTrace}{$D+}{$ENDC}
26 545 --      {$S MADEBUG}
26 546 --      {-----+
26 547 --      |      PointerToHex
26 548 --      +-----+}
26 549 -- A  PROCEDURE PointerToHex (theNumber: LONGINT; VAR hexString: Str255; hexDigits: INTEGER);
26 550 --
26 551 --      VAR
26 552 --          tempString:      String8;
26 553 --
26 554 0- A  BEGIN
26 555 --          IF theNumber = 0 THEN
26 556 --              hexString := 'Nil'
26 557 --          ELSE
26 558 1-      BEGIN
26 559 --              LIntToHex(theNumber, tempString, hexDigits);
26 560 --              hexString := CONCAT('$', tempString);
26 561 1-      END;
26 562 0- A  END {PointerToHex};
26 563 --      {$IFC qTrace}{$D+}{$ENDC}
26 564 --
26 565 --      {$S MAFile}
26 566 --      {-----+
26 567 --      |      NumBlocks
26 568 --      +-----+}
26 569 -- A  FUNCTION NumBlocks (numBytes: LONGINT; blkSize: LONGINT): LONGINT;
26 570 --
26 571 0- A  BEGIN
26 572 --          NumBlocks := (numBytes + blkSize - 1) DIV blkSize;
26 573 0- A  END {NumBlocks};
26 574 --
26 575 --      {$S MAFile}
26 576 --      {-----+
26 577 --      |      OpenFile
26 578 --      +-----+}
26 579 -- A  FUNCTION OpenFile(name: Str255; volRefnum: INTEGER; openData, openRsrc: BOOLEAN;
26 580 --                      dataPerm, rsrcPerm: INTEGER;
26 581 --                      VAR dataRefnum, rsrcRefnum: INTEGER): OSErr;
26 582 --
26 583 --      VAR
26 584 --          pb:      HParamBlockRec;
26 585 --          oldVRefnum: INTEGER;
26 586 --          result:   OSErr;
26 587 --
26 588 --      {-----+
26 589 --      |      TestForError
26 590 --      +-----+}
26 590 -- B  PROCEDURE TestForError (err: OSErr);
26 591 --
26 592 0- B  BEGIN
26 593 --      IF err <> noErr THEN

```

```

26 594 1-      BEGIN
26 595 --      OpenFile := err;
26 596 --      EXIT(OpenFile);
26 597 -1      END;
26 598 -0 B      END (TestForError);
26 599 --
26 600 0- A      BEGIN
26 601 --          {always open data fork, to establish that the file does exist}
26 602 --          WITH pb DO
26 603 1-              BEGIN
26 604 --                  ioNamePtr := @name;
26 605 --                  ioVRefNum := volRefnum;
26 606 --                  ioVersNum := 0;
26 607 --                  ioPermssn := dataPerm;
26 608 --                  ioMisc := NIL;
26 609 -1              END;
26 610 --          TestForError(FillInDirID(@pb));
26 611 --
26 612 --          IF gConfiguration.hasHFS THEN
26 613 --              result := PBHOpenDeny(@pb, FALSE)          ( Try the shared volume open. )
26 614 --          ELSE
26 615 --              result := paramErr;
26 616 --          IF result = paramErr THEN                      ( Not on a shared volume, try HFS open. )
26 617 1-              BEGIN
26 618 --                  pb.ioPermssn := BAND(dataPerm, 3);
26 619 --                  result := PBHOpen(@pb, FALSE);
26 620 -1              END;
26 621 --          TestForError(result);
26 622 --
26 623 --          IF openData THEN
26 624 --              dataRefnum := pb.ioRefnum
26 625 --          ELSE
26 626 1-              BEGIN
26 627 --                  { we did not want the data fork open, so close it now }
26 628 --                  TestForError(FSClose(pb.ioRefnum));
26 629 --                  dataRefnum := kNoFileRefnum;
26 630 -1              END;
26 631 --
26 632 --          IF openRsrc THEN
26 633 1-              BEGIN
26 634 --                  {$IFC NOT qNeedsROM128K}
26 635 --                      IF NOT gConfiguration.hasROM128K THEN
26 636 2-                          BEGIN
26 637 --                              TestForError(GetVol(NIL, oldVRefnum));
26 638 --                              TestForError(SetVol(NIL, volRefnum));
26 639 --
26 640 --                              rsrcRefnum := OpenResFile(name);
26 641 --
26 642 --                              TestForError(SetVol(NIL, oldVRefnum));
26 643 -2                          END
26 644 --                      ELSE
26 645 --                          {$ENDC}
26 646 2-                          BEGIN
26 647 --                              IF gConfiguration.hasHFS THEN
26 648 --                                  result := PBHOpenRFDeny(@pb, FALSE)
26 649 --                              ELSE
26 650 --                                  result := paramErr;
26 651 --                              IF result = paramErr THEN
26 652 3-                                  BEGIN
26 653 --                                      rsrcRefnum := OpenRFPPerm(name, volRefnum, BAND(rsrcPerm, 7));
26 654 --                                      result := ResError;
26 655 -3                                  END
26 656 --                                  ELSE
26 657 --                                      rsrcRefnum := pb.ioRefnum;
26 658 -2                                  END;
26 659 --
26 660 --                                  IF result <> noErr THEN
26 661 --                                      rsrcRefnum := kNoFileRefnum;
26 662 -1                                  END
26 663 --                                  ELSE
26 664 --                                      rsrcRefnum := kNoFileRefnum;
26 665 --
26 666 --                                      OpenFile := noErr;
26 667 --
26 668 -0 A          END (OpenFile);
26 669 --
26 670 --          {$S MAREs}
26 671 --          {-----+
26 672 --          | PinOnRect |
26 673 --          +-----+}
26 674 -- A      FUNCTION PinOnRect(theRect: Rect; thePt: Point): LONGINT;
26 675 --
26 676 0- A      BEGIN
26 677 --          IF thePt.h < theRect.left THEN thePt.h := theRect.left;
26 678 --          IF thePt.h > theRect.right THEN thePt.h := theRect.right;
26 679 --          IF thePt.v < theRect.top THEN thePt.v := theRect.top;
26 680 --          IF thePt.v > theRect.bottom THEN thePt.v := theRect.bottom;
26 681 --
26 682 --          PinOnRect := LONGINT(thePt);
26 683 -0 A      END (PinOnRect);
26 684 --
26 685 --          {$IFC qDebug}
26 686 --          {$S MADebug}
26 687 --          {-----+
26 688 --          | ReadInteger |
26 689 --          +-----+}
26 690 -- A      FUNCTION ReadInteger (prompt: Str255): INTEGER;
26 691 --
26 692 --      VAR

```

```

26 693 --      i:      INTEGER;
26 694 --
26 695 0- A BEGIN
26 696 --      WWForceOutput(forceOn, forceUnchanged);
26 697 --      Write(prompt);
26 698 --      Readln(i);
26 699 --      WWEndForce;
26 700 --      ReadInteger := i;
26 701 -0 A END {ReadInteger};
26 702 --      {$ENDC qDebug}
26 703 --
26 704 --      {$IFC qDebug}
26 705 --      {$S MAdDebug}
26 706 --      {-----+
26 707 --      |   ReadYesNo
26 708 --      +-----}
26 709 -- A FUNCTION ReadYesNo (prompt: Str255): BOOLEAN;
26 710 --
26 711 --      VAR
26 712 --          s:      Str255;
26 713 --
26 714 0- A BEGIN
26 715 --      WWForceOutput(forceOn, forceUnchanged);
26 716 --      Write(prompt);
26 717 --      Readln(s);
26 718 --      WWEndForce;
26 719 --      ReadYesNo := (s <> '') & (s[1] IN ['y', 'Y']);
26 720 -0 A END {ReadYesNo};
26 721 --      {$ENDC qDebug}
26 722 --
26 723 --      {$S MRes}
26 724 --      {-----+
26 725 --      |   RectsNest
26 726 --      +-----}
26 727 -- A FUNCTION RectsNest(outer, inner: Rect): BOOLEAN;
26 728 0- A BEGIN
26 729 --      WITH inner DO
26 730 --          RectsNest := (left >= outer.left) AND (right <= outer.right)
26 731 --                      AND (top >= outer.top) AND (bottom <= outer.bottom);
26 732 -0 A END;
26 733 --
26 734 --      {$S MRes}
26 735 --      {-----+
26 736 --      |   RoundUp
26 737 --      +-----}
26 738 -- A FUNCTION RoundUp(aNumber: LONGINT; aModulus: INTEGER): LONGINT;
26 739 --
26 740 0- A BEGIN
26 741 --      RoundUp := ((aNumber + aModulus - 1) DIV aModulus) * aModulus;
26 742 -0 A END {RoundUp};
26 743 --
26 744 --      {$S MAMain}
26 745 --      {-----+
26 746 --      |   SetKeyScript
26 747 --      +-----}
26 748 -- A PROCEDURE SetKeyScript (newKeyScript: INTEGER);
26 749 --
26 750 --      VAR
26 751 --          currentKeyScript: INTEGER;
26 752 --
26 753 0- A BEGIN
26 754 --      currentKeyScript := GetEnvirons(smKeyScript);
26 755 --      IF currentKeyScript <> newKeyScript THEN
26 756 --          KeyScript(newKeyScript);
26 757 -0 A END {SetKeyScript};
26 758 --
26 759 --      {$S Main} {Must be in Main segment and cannot call to any other segment.}
26 760 --      {$IFC qTrace}{$D+}{$ENDC}
26 761 --      {-----+
26 762 --      |   SetHandleBits
26 763 --      +-----}
26 764 -- A PROCEDURE SetHandleBits(h: Handle; theBits: SignedByte);
26 765 --
26 766 0- A BEGIN
26 767 --      {$IFC NOT qNeedsROM128K}
26 768 --          IF NOT gConfiguration.hasROM128K THEN
26 769 --              PByte(h)^ := theBits
26 770 --          ELSE
26 771 --              {$ENDC}
26 772 --              HSetState(h, theBits);
26 773 -0 A END;
26 774 --      {$IFC qTrace}{$D+}{$ENDC}
26 775 --
26 776 --      {$S MAMain}
26 777 --      {-----+      Set present foreground color
26 778 --      |   SetIfColor
26 779 --      +-----}      Works with both old and new QuickDraw
26 780 -- A PROCEDURE SetIfColor (aColor: RGBColor);
26 781 --      CONST
26 782 --          SignBit = 15;
26 783 --      VAR
26 784 --          index:      INTEGER;
26 785 --          oldColor:    LONGINT;
26 786 0- A BEGIN
26 787 --      IF gConfiguration.hasColorQD THEN
26 788 --          RGBForeColor(aColor)
26 789 --      ELSE
26 790 1-      BEGIN
26 791 --          index := 0;
26 791 --          { Prime index

```

```

26 792 -- THEN
26 793 --     index := 4;
26 794 --     IF BTST(aColor.green, SignBit) THEN      ( Set bit if green >= $8000
26 795 --         index := index + 2;
26 796 --     IF BTST(aColor.blue, SignBit) THEN      ( Set bit if blue >= $8000
26 797 --         index := index + 1;
26 798 --     CASE index OF
26 799 --         0: oldColor := blackColor;
26 800 --         1: oldColor := blueColor;
26 801 --         2: oldColor := greenColor;
26 802 --         3: oldColor := cyanColor;
26 803 --         4: oldColor := redColor;
26 804 --         5: oldColor := magentaColor;
26 805 --         6: oldColor := yellowColor;
26 806 --         7: oldColor := whiteColor;
26 807 --     END;
26 808 --     ForeColor(oldColor);
26 809 --     END;
26 810 -- A   END;
26 811 --
26 812 --     {$S MAMain}
26 813 --     {-----+
26 814 --     | SetPortTextStyle
26 815 --     +-----}
26 816 -- A   PROCEDURE SetPortTextStyle(theTextStyle: TextStyle);
26 817 --
26 818 -- A   BEGIN
26 819 --       TextFont(theTextStyle.tsFont);
26 820 --       TextFace(theTextStyle.tsFace);
26 821 --       TextSize(theTextStyle.tsSize);
26 822 --       SetIfColor(theTextStyle.tsColor);
26 823 -- A   END {SetPortTextStyle};
26 824 --
26 825 --     {$S MAMain}
26 826 --     {-----+
26 827 --     | SetRGBColor
26 828 --     +-----}
26 829 -- A   PROCEDURE SetRGBColor (VAR RGB: RGBColor; red, green, blue: INTEGER);
26 830 --
26 831 -- A   BEGIN
26 832 --       RGB.red := red;
26 833 --       RGB.green := green;
26 834 --       RGB.blue := blue;
26 835 -- A   END;
26 836 --
26 837 --     {$S MAMain}
26 838 --     {-----+
26 839 --     | SetTextStyle
26 840 --     +-----} Convenience routine ala SetRect
26 841 -- A   PROCEDURE SetTextStyle(VAR theTextStyle: TextStyle; theFont: INTEGER;
26 842 --       theStyle: Style; theSize: INTEGER; theColor: RGBColor);
26 843 -- A   BEGIN
26 844 --       WITH theTextStyle DO
26 845 --       BEGIN
26 846 --         tsFont := theFont;
26 847 --         tsFace := theStyle;
26 848 --         tsSize := theSize;
26 849 --         tsColor := theColor;
26 850 --       END;
26 851 -- A   END;
26 852 --
26 853 --     {$IFC qDebug}
26 854 --     {$IFC qTrace}{$D+}{$ENDC}
26 855 --     {$S MAMain}
26 856 --     {-----+
26 857 --     | StdFieldToString
26 858 --     +-----}
26 859 -- A   PROCEDURE StdFieldToString (theData: Ptr; fieldType: INTEGER; VAR theString: Str255);
26 860 --
26 861 --     CONST
26 862 --         adnFrame = [adnLineTop, adnLineLeft, adnLineBottom, adnLineRight];
26 863 --
26 864 --     TYPE
26 865 --         Talias = RECORD
26 866 --             CASE INTEGER OF
26 867 --                 bBoolean: (asBoolean: BOOLEAN);
26 868 --                 bFontName,
26 869 --                 bCmdNumber,
26 870 --                 bHighByte,
26 871 --                 bLowByte,
26 872 --                 bHexInteger,
26 873 --                 bInteger: (asInteger: INTEGER);
26 874 --                 bFixed,
26 875 --                 bHexLongInt,
26 876 --                 bLongInt: (asLongInt: LONGINT);
26 877 --                 bString: (asString: Str255);
26 878 --                 bChar: (asChar: CHAR);
26 879 --                 bGrafPtr,
26 880 --                 bWindowPtr,
26 881 --                 bPointer: (asPointer: Ptr);
26 882 --                 bRgnHandle,
26 883 --                 bControlHandle,
26 884 --                 bTEHandle,
26 885 --                 bHandle: (asHandle: Handle);
26 886 --                 bPoint: (asPoint: Point);
26 887 --                 bRect: (asRect: Rect);
26 888 --                 bObject: (asObject: Handle);
26 889 --                 bByte: (asByte: SignedByte);
26 890 --                 bOSType: (asOSType: OSType);

```

```

26 891 --      bPattern:      (asPattern:      Pattern);
26 892 --      bRGBColor:     (asRGBColor:     RGBColor);
26 893 --      bStyle:        (asStyle:        Style);
26 894 --      bVCoordinate:  (asVCoordinate:  VCoordinate);
26 895 --      bVPoint:       (asVPoint:       VPoint);
26 896 --      bVRect:        (asVRect:        VRect);
26 897 --      bStringHandle: (asStrHandle:    StringHandle);
26 898 --      bCntlAdornment: (asCntlAdornment: CntlAdornment);
26 899 --      END;
26 900 --
26 901 --      VAR
26 902 --      alias:      ^TAlias;
26 903 --      aString:    Str255;
26 904 --      hexString:  String8;
26 905 --      i:          INTEGER;
26 906 --
26 907 --      {-----+
26 908 --      | CheckStyleItem |
26 909 --      +-----+}
26 910 -- B  PROCEDURE CheckStyleItem (s: StyleItem; name: Str255);
26 911 --
26 912 -- B      BEGIN
26 913 --      IF s IN alias^.asStyle THEN
26 914 --      IF theString = '[' THEN
26 915 --      theString := CONCAT(theString, name)
26 916 --      ELSE
26 917 --      theString := CONCAT(theString, ',', name);
26 918 -- B      END;
26 919 --
26 920 --      {-----+
26 921 --      | CheckAdornment |
26 922 --      +-----+}
26 923 -- B  PROCEDURE CheckAdornment (p: CntlAdornment; name: Str255);
26 924 --
26 925 -- B      BEGIN
26 926 --      { "set1 <= set2" means set1 is wholly contained in set2 }
26 927 --      IF p <= alias^.asCntlAdornment THEN
26 928 --      IF theString = '[' THEN
26 929 --      theString := CONCAT(theString, name)
26 930 --      ELSE
26 931 --      theString := CONCAT(theString, ',', name);
26 932 -- B      END;
26 933 --
26 934 -- A  BEGIN
26 935 --      alias := Pointer(theData);
26 936 --      theString := '';
26 937 --      WITH alias^ DO
26 938 --      CASE fieldType OF
26 939 --      bBoolean:
26 940 --      theString := gBoolString[ORD(asBoolean) <> 0];
26 941 --      bFontName:
26 942 --      GetFontName(asInteger, theString);
26 943 --      bInteger:
26 944 --      NumToString(asInteger, theString);
26 945 --      bLongInt:
26 946 --      NumToString(asLongInt, theString);
26 947 --      bHexInteger:
26 948 --      NumberToHex(asInteger, theString, 4);
26 949 --      bHexLongInt:
26 950 --      NumberToHex(asLongInt, theString, 8);
26 951 --      bHighByte:
26 952 --      NumberToHex(BAND(asInteger, $FF00), theString, 2);
26 953 --      bLowByte:
26 954 --      NumberToHex(BAND(asInteger, $00FF), theString, 2);
26 955 --      bFixed:
26 956 --      BEGIN
26 957 --      NumToString(HiWrd(asLongInt), aString);
26 958 --      NumToString(LoWrd(asLongInt), theString);
26 959 --      theString := CONCAT(aString, ':', theString);
26 960 --      END;
26 961 --      bString:
26 962 --      theString := asString;
26 963 --      bChar:
26 964 --      BEGIN
26 965 --      theString := ' ';
26 966 --      theString[1] := asChar;
26 967 --      END;
26 968 --      bGrafPtr,
26 969 --      bWindowPtr,
26 970 --      bPointer:
26 971 --      PointerToHex(ORD4(asPointer), theString, 6);
26 972 --      bRgnHandle,
26 973 --      bControlHandle,
26 974 --      bTEHandle,
26 975 --      bHandle:
26 976 --      IF asHandle = NIL THEN
26 977 --      theString := 'Nil'
26 978 --      ELSE
26 979 --      NumberToHex(ORD4(asHandle), theString, 6);
26 980 --      bPoint:
26 981 --      BEGIN
26 982 --      NumToString(asPoint.h, aString);
26 983 --      NumToString(asPoint.v, theString);
26 984 --      theString := CONCAT('(', aString, ', ', theString, ')');
26 985 --      END;
26 986 --      bRect:
26 987 --      BEGIN
26 988 --      NumToString(asRect.left, aString);
26 989 --      NumToString(asRect.top, theString);

```

```

26 990 --      theString := CONCAT('(', aString, ', ', theString, ')/');
26 991 --      NumToString(asRect.right, aString);
26 992 --      theString := CONCAT(theString, aString, ', ');
26 993 --      NumToString(asRect.bottom, aString);
26 994 --      theString := CONCAT(theString, aString, '));
26 995 --      END;
26 996 --      bObject:
26 997 --      PointerToHex(ORD4(asObject), theString, 6);
26 998 --      bByte:
26 999 --      NumToString(asByte, theString);
26 1000 --      bCmdNumber:
26 1001 --      NumToString(asInteger, theString);
26 1002 --      bOSType:
26 1003 --      BEGIN
26 1004 --      theString := ' ';
26 1005 --      FOR i := 1 TO 4 DO
26 1006 --      theString[i + 1] := asOSType[i];
26 1007 --      END;
26 1008 --      bPattern:
26 1009 --      BEGIN
26 1010 --      theString := '$';
26 1011 --      FOR i := 0 TO 7 DO
26 1012 --      BEGIN
26 1013 --      LIntToHex(asPattern[i], hexString, 2);
26 1014 --      theString := CONCAT(theString, hexString);
26 1015 --      END;
26 1016 --      END;
26 1017 --      bRGBColor:
26 1018 --      WITH asRGBColor DO
26 1019 --      IF (red = 0) AND (green = 0) AND (blue = 0) THEN
26 1020 --      theString := 'Black'
26 1021 --      ELSE IF (red = $FFFF) AND (green = $FFFF) AND (blue = $FFFF) THEN
26 1022 --      theString := 'White'
26 1023 --      ELSE
26 1024 --      BEGIN
26 1025 --      NumberToHex(asRGBColor.red, theString, 4);
26 1026 --      NumberToHex(asRGBColor.green, aString, 4);
26 1027 --      theString := CONCAT(theString, '/', aString);
26 1028 --      NumberToHex(asRGBColor.blue, aString, 4);
26 1029 --      theString := CONCAT(theString, '/', aString);
26 1030 --      END;
26 1031 --      bStyle:
26 1032 --      BEGIN
26 1033 --      theString := '[';
26 1034 --      CheckStyleItem(bold, 'bold');
26 1035 --      CheckStyleItem(italic, 'italic');
26 1036 --      CheckStyleItem(underline, 'underline');
26 1037 --      CheckStyleItem(outline, 'outline');
26 1038 --      CheckStyleItem(shadow, 'shadow');
26 1039 --      CheckStyleItem(condense, 'condense');
26 1040 --      CheckStyleItem(extend, 'extend');
26 1041 --      theString := CONCAT(theString, ']');
26 1042 --      END;
26 1043 --      bVCoordinate:
26 1044 --      NumToString(asVCoordinate, theString);
26 1045 --      bVPoint:
26 1046 --      BEGIN
26 1047 --      NumToString(asVPoint.h, aString);
26 1048 --      NumToString(asVPoint.v, theString);
26 1049 --      theString := CONCAT('(', aString, ', ', theString, ')/');
26 1050 --      END;
26 1051 --      bVRect:
26 1052 --      BEGIN
26 1053 --      NumToString(asVRect.left, aString);
26 1054 --      NumToString(asVRect.top, theString);
26 1055 --      theString := CONCAT('(', aString, ', ', theString, ')/');
26 1056 --      NumToString(asVRect.right, aString);
26 1057 --      theString := CONCAT(theString, aString, ', ');
26 1058 --      NumToString(asVRect.bottom, aString);
26 1059 --      theString := CONCAT(theString, aString, '));
26 1060 --      END;
26 1061 --      bStringHandle:
26 1062 --      IF asStrHandle = NIL THEN
26 1063 --      theString := 'Nil'
26 1064 --      ELSE
26 1065 --      theString := asStrHandle^^;
26 1066 --      bCntlAdornment:
26 1067 --      BEGIN
26 1068 --      theString := '[';
26 1069 --      IF adnFrame <= alias^.asCntlAdornment THEN
26 1070 --      CheckAdornment(adnFrame, 'frame')
26 1071 --      ELSE
26 1072 --      BEGIN
26 1073 --      CheckAdornment([adnLineTop], 'top');
26 1074 --      CheckAdornment([adnLineLeft], 'left');
26 1075 --      CheckAdornment([adnLineBottom], 'bottom');
26 1076 --      CheckAdornment([adnLineRight], 'right');
26 1077 --      END;
26 1078 --      (* CheckAdornment(adnPatFill, 'fill'); *)
26 1079 --      CheckAdornment([adnOval], 'oval');
26 1080 --      CheckAdornment([adnRRect], 'rrect');
26 1081 --      CheckAdornment([adnShadow], 'shadow');
26 1082 --      theString := CONCAT(theString, ']');
26 1083 --      END;
26 1084 --      END;
26 1085 --      A END {StdFieldToString};
26 1086 --      {$IFC qTrace}{$D++}{$ENDC}
26 1087 --      {$ENDC qDebug}
26 1088 --

```



```

26 1089 --      {$IFC qDebug}
26 1090 --      {$SS MAFields}          { Must be in MAFields as we use this routine to make MAFields resident }
26 1091 --      {-----+
26 1092 --      |   TextStyleFields
26 1093 --      +-----+
26 1094 -- A  PROCEDURE TextStyleFields (aTitle: Str255; VAR aStyle: TextStyle;
26 1095 --                                PROCEDURE DoToField (fieldName: Str255;
26 1096 --                                fieldAddr: Ptr;
26 1097 --                                fieldType: INTEGER));
26 1098 0- A  BEGIN
26 1099 --      DoToField(aTitle, NIL, bTitle);
26 1100 --      DoToField(' Font', @aStyle.tsFont, bFontName);
26 1101 --      DoToField(' Face', @aStyle.tsFace, bStyle);
26 1102 --      DoToField(' Size', @aStyle.tsSize, bInteger);
26 1103 --      DoToField(' Color', @aStyle.tsColor, bRGBColor);
26 1104 -- A  END;
26 1105 --      {$ENDC}
26 1106 --
26 1107 --      {$IFC qTrace}{$D+}{$ENDC}
26 1108 --      {$SS MAMain}
26 1109 --      {-----+
26 1110 --      |   UprChar
26 1111 --      +-----+
26 1112 -- A  FUNCTION UprChar (ch: CHAR): CHAR;
26 1113 --
26 1114 0- A  BEGIN
26 1115 --      IF (ch >= 'a') AND (ch <= 'z') THEN
26 1116 --          UprChar := CHR(ORD(ch) - 32)
26 1117 --      ELSE
26 1118 --          UprChar := ch;
26 1119 0- A  END {UprChar};
26 1120 --      {$IFC qTrace}{$D+}{$ENDC}
26 1121 --
26 1122 --      {$IFC qTrace}{$D+}{$ENDC}
26 1123 --      {$SS MAMain}
26 1124 --      {-----+
26 1125 --      |   UprString8
26 1126 --      +-----+
26 1127 -- A  PROCEDURE UprString8 (VAR s: String8);
26 1128 --
26 1129 --      VAR
26 1130 --          i:      INTEGER;
26 1131 --
26 1132 0- A  BEGIN
26 1133 --      FOR i := 1 TO LENGTH(s) DO
26 1134 --          IF (s[i] IN ['a'..'z']) THEN
26 1135 --              s[i] := CHR(ORD(s[i]) - 32)
26 1136 0- A  END {UprString8};
26 1137 --      {$IFC qTrace}{$D+}{$ENDC}
26 1138 --
26 1139 --      {$SS MAMain}
26 1140 --      {$IFC qTrace}{$D+}{$ENDC}
26 1141 --      {-----+
26 1142 --      |   UseROMMap
26 1143 --      +-----+
26 1144 -- A  PROCEDURE UseROMMap(resLoad: BOOLEAN);
26 1145 --      CONST      mapTrue =      $FFFF;
26 1146 --                  mapFalse =    $FF00;
26 1147 --                  ROMMapInsert = $0B9E;
26 1148 --
26 1149 --      VAR p: PInteger;
26 1150 0- A  BEGIN
26 1151 --      {$IFC NOT qNeedsROM128K}
26 1152 --          IF NOT gConfiguration.hasROM128K THEN
26 1153 --              SetResLoad(resLoad)
26 1154 --          ELSE
26 1155 --              {$ENDC}
26 1156 1-      BEGIN
26 1157 --          p := PInteger(ROMMapInsert);
26 1158 --          IF resLoad THEN
26 1159 --              p^ := mapTrue
26 1160 --          ELSE
26 1161 --              p^ := mapFalse;
26 1162 1-      END;
26 1163 0- A  END {UseROMMap};
26 1164 --      {$IFC qTrace}{$D+}{$ENDC}
26 1165 --
26 1166 --      {$IFC qDebug}
26 1167 --      {$IFC qTrace}{$D+}{$ENDC}
26 1168 --      {$SS MAMain}
26 1169 --      {-----+
26 1170 --      |   WritePt
26 1171 --      +-----+
26 1172 -- A  PROCEDURE WritePt (pt: Point);
26 1173 --
26 1174 0- A  BEGIN
26 1175 --      Write('(', pt.h:1, ', ', pt.v:1, ')');
26 1176 0- A  END {WritePt};
26 1177 --
26 1178 --      {$SS MAMain}
26 1179 --      {-----+
26 1180 --      |   WrLblPt
26 1181 --      +-----+
26 1182 -- A  PROCEDURE WrLblPt(aLabel: Str255; pt: Point);
26 1183 --
26 1184 0- A  BEGIN
26 1185 --      Write(aLabel, ' = '); WritePt(pt);
26 1186 0- A  END {WrLblPt};
26 1187 --      {$IFC qTrace}{$D+}{$ENDC}

```

```

26 1188 --
26 1189 --      {$IFC qTrace}{$D+}{$ENDC}
26 1190 --      {$S MADEbug}
26 1191 --      {-----+
26 1192 --      | WritePtr
26 1193 --      +-----}
26 1194 -- A  PROCEDURE WritePtr (val: UNIV LONGINT);
26 1195 --
26 1196 --      VAR
26 1197 --          hex:   String8;
26 1198 --
26 1199 0- A  BEGIN
26 1200 --          LintToHex(val, hex, 6);
26 1201 --          Write('$ ', hex);
26 1202 -0 A  END (WritePtr);
26 1203 --
26 1204 --      {$S MADEbug}
26 1205 --      {-----+
26 1206 --      | WrLblPtr
26 1207 --      +-----}
26 1208 -- A  PROCEDURE WrLblPtr (aLabel: Str255; val: UNIV LONGINT);
26 1209 --
26 1210 0- A  BEGIN
26 1211 --          Write(aLabel, ' = '); WritePtr(val);
26 1212 -0 A  END (WrLblPtr);
26 1213 --      {$IFC qTrace}{$D+}{$ENDC}
26 1214 --
26 1215 --      {$IFC qTrace}{$D+}{$ENDC}
26 1216 --      {$S MADEbug}
26 1217 --      {-----+
26 1218 --      | WriteRect
26 1219 --      +-----}
26 1220 -- A  PROCEDURE WriteRect (r: Rect);
26 1221 --
26 1222 0- A  BEGIN
26 1223 --          WritePt(r.topLeft);
26 1224 --          Write('/');
26 1225 --          WritePt(r.botRight);
26 1226 -0 A  END (WriteRect);
26 1227 --
26 1228 --      {$S MADEbug}
26 1229 --      {-----+
26 1230 --      | WrLblRect
26 1231 --      +-----}
26 1232 -- A  PROCEDURE WrLblRect (aLabel: Str255; r: Rect);
26 1233 --
26 1234 0- A  BEGIN
26 1235 --          Write(aLabel, ' = '); WriteRect(r);
26 1236 -0 A  END (WrLblRect);
26 1237 --      {$IFC qTrace}{$D+}{$ENDC}
26 1238 --
26 1239 --      {$IFC qTrace}{$D+}{$ENDC}
26 1240 --      {$S MADEbug}
26 1241 --      {-----+
26 1242 --      | WriteBoolean
26 1243 --      +-----}
26 1244 -- A  PROCEDURE WriteBoolean (b: BOOLEAN);
26 1245 --
26 1246 0- A  BEGIN
26 1247 --          IF b THEN
26 1248 --              Write('TRUE')
26 1249 --          ELSE
26 1250 --              Write('FALSE');
26 1251 -0 A  END (WriteBoolean);
26 1252 --
26 1253 --      {$S MADEbug}
26 1254 --      {-----+
26 1255 --      | WrLblBoolean
26 1256 --      +-----}
26 1257 -- A  PROCEDURE WrLblBoolean (aLabel: Str255; b: BOOLEAN);
26 1258 --
26 1259 0- A  BEGIN
26 1260 --          Write(aLabel);
26 1261 --          WriteBoolean(b);
26 1262 -0 A  END (WrLblBoolean);
26 1263 --      {$IFC qTrace}{$D+}{$ENDC}
26 1264 --
26 1265 --      {$IFC qTrace}{$D+}{$ENDC}
26 1266 --      {$S MADEbug}
26 1267 --      {-----+
26 1268 --      | WriteVPt
26 1269 --      +-----}
26 1270 -- A  PROCEDURE WriteVPt (pt: VPoint);
26 1271 --
26 1272 0- A  BEGIN
26 1273 --          Write('(', pt.h:1, ', ', pt.v:1, ')');
26 1274 -0 A  END (WriteVPt);
26 1275 --
26 1276 --      {$S MADEbug}
26 1277 --      {-----+
26 1278 --      | WrLblVPt
26 1279 --      +-----}
26 1280 -- A  PROCEDURE WrLblVPt(aLabel: Str255; pt: VPoint);
26 1281 --
26 1282 0- A  BEGIN
26 1283 --          Write(aLabel, ' = '); WriteVPt(pt);
26 1284 -0 A  END (WrLblVPt);
26 1285 --      {$IFC qTrace}{$D+}{$ENDC}
26 1286 --

```

```

26 1287 --      {$IFC qTrace}{$D+}{$ENDC}
26 1288 --      {$S MADEbug}
26 1289 --      {-----+
26 1290 --      |   WriteVRect
26 1291 --      +-----+}
26 1292 -- A  PROCEDURE WriteVRect (r: VRect);
26 1293 --
26 1294 -- A  BEGIN
26 1295 --      WriteVpt(r.topLeft);
26 1296 --      Write('/');
26 1297 --      WriteVpt(r.botRight);
26 1298 -- A  END {WriteVRect};
26 1299 --
26 1300 --      {$S MADEbug}
26 1301 --      {-----+
26 1302 --      |   WrLblVRect
26 1303 --      +-----+}
26 1304 -- A  PROCEDURE WrLblVRect (aLabel: Str255; r: VRect);
26 1305 --
26 1306 -- A  BEGIN
26 1307 --      Write(aLabel, ' = '); WriteVRect(r);
26 1308 -- A  END {WrLblVRect};
26 1309 --      {$IFC qTrace}{$D+}{$ENDC}
26 1310 --
26 1311 --      {$IFC qTrace}{$D+}{$ENDC}
26 1312 --      {$S MADEbug}
26 1313 --      {-----+
26 1314 --      |   WriteSig
26 1315 --      +-----+}
26 1316 -- A  PROCEDURE WriteSig(theID: IDType);
26 1317 --      VAR
26 1318 --          theSig: STRING[4];
26 1319 -- A  BEGIN
26 1320 --      theSig := '....';
26 1321 --      IF LONGINT(theID) <> 0 THEN
26 1322 --          BlockMove(@theID, Ptr(ORD4(@theSig)+1), SIZEOF(IDType));
26 1323 --          Write('...', theSig, '');
26 1324 -- A  END;
26 1325 --
26 1326 --      {$S MADEbug}
26 1327 --      {-----+
26 1328 --      |   WrLblSig
26 1329 --      +-----+}
26 1330 -- A  PROCEDURE WrLblSig(theLabel: Str255; theID: IDType);
26 1331 -- A  BEGIN
26 1332 --      Write(theLabel, ' = '); WriteSig(theID);
26 1333 -- A  END;
26 1334 --      {$IFC qTrace}{$D+}{$ENDC}
26 1335 --
26 1336 --      {$IFC qTrace}{$D+}{$ENDC}
26 1337 --      {$S MADEbug}
26 1338 --      {-----+
26 1339 --      |   WriteHexInt
26 1340 --      +-----+}
26 1341 -- A  PROCEDURE WriteHexInt(theInt: INTEGER);
26 1342 --      VAR
26 1343 --          tempString:   String8;
26 1344 -- A  BEGIN
26 1345 --      LIntToHex(theInt, tempString, 4);
26 1346 --      Write('$ ', tempString);
26 1347 -- A  END {WriteHexInt};
26 1348 --
26 1349 --      {$S MADEbug}
26 1350 --      {-----+
26 1351 --      |   WrLblHexInt
26 1352 --      +-----+}
26 1353 -- A  PROCEDURE WrLblHexInt(theLabel: Str255; theInt: INTEGER);
26 1354 -- A  BEGIN
26 1355 --      Write(theLabel, ' = '); WriteHexInt(theInt);
26 1356 -- A  END;
26 1357 --      {$IFC qTrace}{$D+}{$ENDC}
26 1358 --
26 1359 --      {$IFC qTrace}{$D+}{$ENDC}
26 1360 --      {$S MADEbug}
26 1361 --      {-----+
26 1362 --      |   WriteHexLongint
26 1363 --      +-----+}
26 1364 -- A  PROCEDURE WriteHexLongint(theLongint: LONGINT);
26 1365 --      VAR
26 1366 --          tempString:   String8;
26 1367 -- A  BEGIN
26 1368 --      LIntToHex(theLongint, tempString, 8);
26 1369 --      Write('$ ', tempString);
26 1370 -- A  END;
26 1371 --
26 1372 --      {$S MADEbug}
26 1373 --      {-----+
26 1374 --      |   WrLblHexLongint
26 1375 --      +-----+}
26 1376 -- A  PROCEDURE WrLblHexLongint(theLabel: Str255; theLongint: LONGINT);
26 1377 -- A  BEGIN
26 1378 --      Write(theLabel, ' = '); WriteHexLongint(theLongint);
26 1379 -- A  END;
26 1380 --      {$IFC qTrace}{$D+}{$ENDC}
26 1381 --      {$ENDC qDebug}
25 505 --
25 506 --      END { UMAUtil }.

```

```

27 1 --      ($P)
27 2 --      { UMemory.p }
27 3 --      { Copyright 1985-1988 by Apple Computer, Inc. All rights reserved. }
27 4 --
27 5 --      { Conditional compilation flags used by this unit:
27 6 --
27 7 --          qNeedsROM128K    True skips code for Mac XL and 64K ROMs,
27 8 --                          false leaves it in.
27 9 --
27 10 --          qDebug          True includes debugging code, false ignores debugging code.
27 11 --                          True also sets $N+.
27 12 --
27 13 --          qTrace          True surrounds trap patches with $D+ and $D++ (because
27 14 --                          the default is $D++), false skips these directives.
27 15 --
27 16 --          The settings of $D, $R and $N are imported from UMAUtil.
27 17 --      }
27 18 --
27 19 --      UNIT UMemory:
27 20 --
27 21 --      {
27 22 --          This unit implements MacApp's memory management and segment management
27 23 --          schemes.
27 24 --
27 25 --          The memory management scheme works by distinguishing between "permanent"
27 26 --          and "temporary" heap allocation requests. Permanent requests are
27 27 --          typically used for data your application allocates: objects and
27 28 --          handles. Temporary requests are used for code segments and Toolbox
27 29 --          resources and data, such as WDEF's and CDEF's.
27 30 --
27 31 --          Permanent memory objects are considered permanent because they
27 32 --          are not purged from memory until you explicitly dispose of or free them.
27 33 --          (Of course, the Macintosh Memory Manager will purge them if they're
27 34 --          marked purgable.) Permanent objects are allocated with NewPermHandle.
27 35 --          This prevents the MacApp GrowZone from purging temporary objects to
27 36 --          accomodate a permanent one. In MacApp, all TObjects and its descendants
27 37 --          are considered permanent, and are allocated with NewPermHandle.
27 38 --
27 39 --          Temporary memory objects are considered temporary because MacApp's
27 40 --          GrowZone procedure may purge them from memory to satisfy a memory
27 41 --          allocation request. This is true regardless of whether the object is
27 42 --          marked non-purgable, although the GrowZone procedure will not purge
27 43 --          locked objects (such as code segments in use). Typically, temporary
27 44 --          objects are marked non-purgable so that MacApp's GrowZone can control
27 45 --          when they are purged.
27 46 --
27 47 --          MacApp reserves a specific amount of heap space for temporary objects,
27 48 --          the idea being that the space reserved is large enough to handle the
27 49 --          largest number of temporary objects (e.g. code and system resources)
27 50 --          needed at any given time. If this amount is sufficiently large, your
27 51 --          program will never fail loading a segment or system resource. This
27 52 --          amount is defined by the internal variable gSzCodeReserve. You can
27 53 --          retrieve its value by calling GetReserveSize. It is initially set by
27 54 --          the 'seg!' and 'mem!' resources, and can be changed at run-time by
27 55 --          calling SetReserveSize. The procedure BuildCodeReserve reserves the
27 56 --          space by allocating the handle gCodeReserve and setting its size to
27 57 --          gSzCodeReserve - (the total size of all temporary objects in memory).
27 58 --
27 59 --          When a temporary object is loaded into memory, the size of gCodeReserve
27 60 --          is adjusted accordingly. Permanent objects can be loaded into memory
27 61 --          only so long as there will still be at least gSzCodeReserve bytes
27 62 --          available for temporary objects.
27 63 --
27 64 --          To identify which objects in the heap are temporary, MacApp maintains
27 65 --          four lists of handles. The objects identified by these handles are
27 66 --          considered temporary and fall under the control of MacApp's GrowZone
27 67 --          procedure. The lists are:
27 68 --
27 69 --          gCodeSegs          A list of handles to all CODE resources in the appli-
27 70 --                             cation and system resource forks.
27 71 --
27 72 --          gSysMemList        A list handles to RAM-based system resources. By
27 73 --                             default this lists includes all PACK, LDEF, MDEF,
27 74 --                             CDEF and WDEF resources in the system and appli-
27 75 --                             cation resource forks. You can add other resources,
27 76 --                             such as fonts, by calling AddHandle.
27 77 --
27 78 --          gApplMemList,      Lists of handles to application data or resources.
27 79 --          gApp2MemList       MacApp initializes both lists to NIL. The difference
27 80 --                             between the two lists is that handles in gApplMemList
27 81 --                             are purged before those in gApp2MemList. You may
27 82 --                             add your own handles to these lists by allocating
27 83 --                             them with NewHandle and calling AddHandle for each
27 84 --                             handle to be added to the list.
27 85 --
27 86 --          The Macintosh Memory Manager calls MacApp's GrowZone procedure only when
27 87 --          all purgable objects have been purged from the heap and there is still
27 88 --          insufficient space to satisfy a memory request. The GrowZone will
27 89 --          look through the lists of temporary objects in memory, making one
27 90 --          purgable so long as there is still at least gSzCodeReserve bytes
27 91 --          allocated to temporary memory (the total size of the temporary objects
27 92 --          in memory and the gCodeReserve handle). The GrowZone procedure attempts
27 93 --          to never allow the size of the temporary objects and reserve to fall
27 94 --          below gSzCodeReserve, thereby guaranteeing that space is always avail-
27 95 --          able for code segments and system resources (provided gSzCodeReserve is
27 96 --          large enough to handle your application at its most memory intensive
27 97 --          state).
27 98 --

```

```

27 99 -- The value of gSzCodeReserve is determined at startup-time by adding up
27 100 -- the size of all the segments named in the 'seg!' resources, and adding
27 101 -- the first value of each of the 'mem!' resources. You can derive these
27 102 -- resources by observing your program and using the MacApp debugger to
27 103 -- help you determine when your application uses the largest amount of
27 104 -- code and system resources. Typically, we've found that this happens
27 105 -- while printing on the LaserWriter, or during initialization or term-
27 106 -- ination.
27 107 --
27 108 -- MacApp maintains another reserve, known as the low memory reserve. This
27 109 -- is a kind of emergency reserve--when all else fails we release the
27 110 -- gMemReserve handle. Its size is initially determined by the second
27 111 -- number of each 'mem!' resource, and can be changed by call
27 112 -- SetReserveSize. You can retrieve its size by calling GetReserveSize.
27 113 -- Internally, the low-memory reserve is allocated with the gMemReserve
27 114 -- handle, and its size is stored in gSzMemReserve.
27 115 --
27 116 -- The function InitUMemory is responsible for initially setting up the
27 117 -- temporary and low-memory reserves, setting up the GrowZone procedure,
27 118 -- initializing handle lists, and patching LoadSeg. It returns true if
27 119 -- the temporary reserve could be allocated.
27 120 --
27 121 -- }
27 122 --
27 123 -- INTERFACE
27 124 --
27 125 -- {-----+
27 126 -- | Uses |
27 127 -- +-----+
27 128 -- USES
27 129 --     {$LOAD MacIntf.LOAD}
27 130 --     MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
27 131 --     {$LOAD}
27 132 --     UMAUtil,
27 133 --     UFailure,
27 134 --     UPatch
27 135 --     {$IFC qDebug}
27 136 --         , UTrace
27 137 --     {$ENDC}
27 138 --     ;
27 139 --
27 140 -- {-----+
27 141 -- | Global constants |
27 142 -- +-----+
27 143 -- {$IFC NOT qDebug}
27 144 -- CONST
27 145 --     gUnloadAllSegs - TRUE; { Always TRUE when not debugging }
27 146 -- {$ENDC}
27 147 --
27 148 -- {-----+
27 149 -- | Global types |
27 150 -- +-----+
27 151 -- TYPE
27 152 --     HHandleList = ^PHandleList;
27 153 --     PHandleList = ^AHandleList;
27 154 --     AHandleList = ARRAY[1..5000] OF Handle; { A list of handles }
27 155 --
27 156 -- {-----+
27 157 -- | Global variables |
27 158 -- +-----+
27 159 -- VAR
27 160 --     gMaxLockedRsrc: LONGINT;
27 161 --     {$IFC qDebug}
27 162 --     gMemMgtBreak: BOOLEAN; { if TRUE, break into debugger rather than
27 163 --                             just report memory mgt. information }
27 164 --     gRsrcReport: BOOLEAN; { report resource maximums }
27 165 --     gSegReport: BOOLEAN; { report segment loadings }
27 166 --     {$ENDC}
27 167 --     gSysMemList: HHandleList; { List of system handles used to
27 168 --                               compute current allocated temporary
27 169 --                               memory (all PACK, LDEF, MDEF, CDEF,
27 170 --                               WDEF resources) }
27 171 --     gApp1MemList: HHandleList;
27 172 --     gApp2MemList: HHandleList;
27 173 --     { These start out NIL. You can create lists of handles and place
27 174 --       then in either of these variables. (One list might be permanent
27 175 --       handles in your application, and the other based on the current
27 176 --       situation.) The values stored here should be a handle to a
27 177 --       list of other handles.
27 178 --
27 179 --       If you modify either of these lists, call CheckReserve. It
27 180 --       calls BuildAllReserves to recompute the code and low space
27 181 --       reserve. CheckReserve's result indicates whether the full code
27 182 --       reserve is present. If not, then your program may crash because
27 183 --       a segment (or defproc) can't be loaded.
27 184 --
27 185 --       The handles in the lists should normally be resources that your
27 186 --       application might require at any given time. All these handles
27 187 --       will normally be marked non-purgeable. The GrowZone proc,
27 188 --       however, can purge any of these handles that are not locked.)
27 189 --
27 190 --     gCodeSegs: HHandleList; { List of all code segments }
27 191 --     gNonResidents: HHandleList; { List of non resident segments; a copy
27 192 --                                  of gCodeSegs with resident segments
27 193 --                                  replaced by NIL }
27 194 --     {$IFC qDebug}
27 195 --     gUnloadAllSegs: BOOLEAN; { UnloadAllSegments doesn't do anything
27 196 --                               if this is false. }
27 197 --     {$ENDC}

```

```

27 198 --
27 199 -- {-----+
27 200 -- | Global routines |
27 201 -- +-----}
27 202 -- { I N I T I A L I Z A T I O N }
27 203 --
27 204 -- FUNCTION InitUMemory: BOOLEAN;
27 205 -- { Initializes this unit. Must be called before using this unit. The
27 206 -- called must be in the program's main segment. Sets up the
27 207 -- gCodeSegs and gSysMemList handle lists, sets the grow zone,
27 208 -- calls MaxApplZone, sets gApplMemList and gApp2MemList to NIL,
27 209 -- patches LoadSeg and allocates the temporary and low-memory
27 210 -- reserves.
27 211 --
27 212 -- Returns false if unable to allocate the temporary reserve. }
27 213 --
27 214 --
27 215 -- { S E G M E N T L O A D I N G }
27 216 --
27 217 -- { DEBUGGING NOTE: You cannot set a MacApp breakpoint at any of these
27 218 -- routines, because they must not call anything (eg. %_BP) that may
27 219 -- require a segment load. }
27 220 --
27 221 -- FUNCTION GetSegNumber(aProc: ProcPtr): INTEGER;
27 222 -- { GetSegNumber returns the number of the segment in which aProc resides }
27 223 --
27 224 -- FUNCTION GetSegSize(segnum: INTEGER): Size;
27 225 -- { GetSegSize returns the size, in bytes, of the segment whose number
27 226 -- is segnum. }
27 227 --
27 228 -- FUNCTION LoadMacAppSegment(segNum: INTEGER): LONGINT;
27 229 -- { This function patches LoadSeg. It is called from the assembly-language
27 230 -- routine AMacAppLoadSeg, which is the actual patch setup by PatchTrap.
27 231 -- AMacAppLoadSeg saves D0-D1-A0/A1 and sets up the stack by
27 232 -- 1) allocating space for LoadMacAppSegment's result, and
27 233 -- 2) pushing a copy of LoadSeg's parameter onto the stack for use
27 234 -- by LoadMacAppSegment. }
27 235 --
27 236 -- PROCEDURE LoadResidentSegments;
27 237 -- { Makes resident the segments whose names are included in any 'res!'
27 238 -- resources. }
27 239 --
27 240 -- FUNCTION PreloadSegment(segNum: INTEGER): BOOLEAN;
27 241 -- { PreloadSegment is available to programmers who want to lock a
27 242 -- segment at the top of the heap without having to call a dummy
27 243 -- procedure in that segment. It is also useful for determining
27 244 -- whether a segment can in fact be loaded before you try to
27 245 -- execute code in it. PreloadSegment returns TRUE if the segment
27 246 -- could be loaded, false otherwise. }
27 247 --
27 248 -- PROCEDURE SetResidentSegment(segNum: INTEGER; makeResident: BOOLEAN);
27 249 -- { SetResidentSegment can be used to make a segment resident (or no
27 250 -- longer resident); resident segments will not be unloaded by
27 251 -- UnloadAllSegments; if a segment is made resident, it is also
27 252 -- preloaded. MacApp automatically marks its resident segment as
27 253 -- resident (the one containing the procedure CmdFromMenuItem); you
27 254 -- probably should do UnloadAllSegments before making a segment
27 255 -- resident, to ensure that it is locked at the top of the heap. }
27 256 --
27 257 -- PROCEDURE UnloadAllSegments;
27 258 -- { UnloadAllSegments unloads all segments except the blank segment
27 259 -- or the ones marked resident. }
27 260 --
27 261 --
27 262 -- { H A N D L E L I S T M A N A G E M E N T }
27 263 --
27 264 -- PROCEDURE AddHandle(h: Handle; toList: HHandleList);
27 265 -- { Adds a handle to the list of handles; does not check if the handle
27 266 -- already exists in the list. Simply calls Munger to add to the
27 267 -- front of the list. }
27 268 --
27 269 -- PROCEDURE AddAllRsrc(rType: ResType; toList: HHandleList);
27 270 -- { Adds all the resources of type rType to the list. Filters out all
27 271 -- ROM resources. Calls AddHandle. }
27 272 --
27 273 -- PROCEDURE RemHandle(h: Handle; toList: HHandleList);
27 274 -- { Removes the handle from list by calling Munger. }
27 275 --
27 276 -- PROCEDURE ScanHandles(PROCEDURE DoToHandle(h: Handle));
27 277 -- { Calls DoToHandle for each handle in the lists gCodeSegs, gSysMemList,
27 278 -- gApplMemList and gApp2MemList. This procedure assumes that DoToHandle
27 279 -- does not compact memory. }
27 280 --
27 281 --
27 282 -- { M E M O R Y M A N A G E M E N T }
27 283 --
27 284 -- PROCEDURE BuildAllReserves;
27 285 -- { BuildAllReserves creates the code (temporary memory) and low space
27 286 -- reserves. }
27 287 --
27 288 -- FUNCTION CheckReserve: BOOLEAN;
27 289 -- { Checks to see if the code reserve is OK. Calls BuildAllReserves and
27 290 -- returns true if the full code reserve is present. If this returns
27 291 -- false your application may bomb because a segment or system
27 292 -- resource can't be loaded. }
27 293 --
27 294 -- {SIFC qDebug}
27 295 -- PROCEDURE CheckRsrcUsage;
27 296 -- { Checks to see if the total size of the currently loaded resources

```

```

27 297 --         exceeds the maximum. If so, the new maximum is set. If gRsrcReport
27 298 --         is true, then the new maximum is reported in the debugger window.
27 299 --         If gMemMgtBreak then program execution is stopped. )
27 300 --     {$ENDC qDebug}
27 301 --
27 302 --     {$IFC qDebug}
27 303 --     PROCEDURE DoChangeReserve(alter: BOOLEAN;
27 304 --         VAR codeReserve, codeShort, lowSpaceReserve: LONGINT;
27 305 --         VAR gotCode, gotLowSpace: BOOLEAN);
27 306 --     { Called by the MacApp debugger. Not normally called by an
27 307 --       application. }
27 308 --     {$ENDC qDebug}
27 309 --
27 310 --     PROCEDURE FailNoReserve;
27 311 --     { IF NOT CheckReserve THEN Failure(memFullErr, 0). }
27 312 --
27 313 --     PROCEDURE FailSpaceIsLow;
27 314 --     { IF MemSpaceIsLow THEN Failure(memFullErr, 0). }
27 315 --
27 316 --     PROCEDURE GetReserveSize (VAR szCodeReserve, szMemReserve: SIZE);
27 317 --     { Returns the amount of memory that is to be reserved for the code and
27 318 --       memory reserve. This is not a true indication of whether this
27 319 --       amount of memory has in fact been reserved. Call CheckReserve to
27 320 --       find out if the code reserve could be allocated, and call
27 321 --       MemSpaceIsLow to find out if the low-memory reserve could be
27 322 --       allocated. }
27 323 --
27 324 --     FUNCTION MemSpaceIsLow: BOOLEAN;
27 325 --     { Returns TRUE if the low space reserve is missing.}
27 326 --
27 327 --     FUNCTION NewPermHandle(logicalSize: Size): Handle;
27 328 --     { Allocates a permanent handle; you should call this instead of
27 329 --       NewHandle if allocating some permanent memory. }
27 330 --
27 331 --     FUNCTION PermAllocation(permanent: BOOLEAN): BOOLEAN;
27 332 --     { PermAllocation controls whether subsequent memory allocations
27 333 --       are considered permanent or temporary. Pass TRUE to setup
27 334 --       things for a permanent allocation. Returns the previous
27 335 --       state of the permanent flag. }
27 336 --
27 337 --     PROCEDURE SetReserveSize(forCode, forOther: Size);
27 338 --     { Call this to set the size of the memory reserved for code (temp-
27 339 --       orary) and permanent (low memory) requests. }
27 340 --
27 341 --     FUNCTION TotalTempSize(justLocked: BOOLEAN; VAR canPurge: Handle): Size;
27 342 --     { TotalTempSize returns the total number of bytes of the handles
27 343 --       in the
27 344 --       {Totals up the "appropriate" handles currently in RAM
27 345 --       (or only locked/in use handles if justLocked is TRUE).
27 346 --       canPurge is set to an unlocked handle. Uses ScanHandles. }
27 347 --
27 348 --     {$IFC qDebug}
27 349 --     PROCEDURE WriteReserves;
27 350 --     { WRITELN's the temporary reserve and low-memory reserves in the
27 351 --       debug window. }
27 352 --     {$ENDC}
27 353 --
27 354 --
27 355 --     { U T I L I T I E S }
27 356 --
27 357 --     FUNCTION AddSegSizes(segRsrc: Handle): LONGINT;
27 358 --     { Returns the total size of the code segments whose names are in the
27 359 --       string list segRsrc. }
27 360 --
27 361 --     PROCEDURE SetStackSpace(numBytes: LONGINT);
27 362 --     { Set the stack space to at least numBytes }
27 363 --
27 364 --
27 365 --     IMPLEMENTATION
27 366 --
27 367 --     {$I UMemory.incl.p}

```

```

28 1 -- ($P)
28 2 -- { UMemory.incl.p }
28 3 -- { Copyright 1985-1988 by Apple Computer, Inc. All rights reserved. }
28 4 --
28 5 -- {-----+
28 6 -- | Local constants |
28 7 -- +-----}
28 8 -- CONST
28 9 --     kGZMaxAlloc    =    $7FFFFFFF;
28 10 --
28 11 -- {-----+
28 12 -- | Local types |
28 13 -- +-----}
28 14 -- TYPE
28 15 --     HBoolArray =    ^PBoolArray;
28 16 --     PBoolArray =    ^BoolArray;
28 17 --     BoolArray =    ARRAY [1..1000] OF BOOLEAN; { Intended to be dynamic }
28 18 --
28 19 -- {-----+
28 20 -- | Local variables |
28 21 -- +-----}
28 22 -- VAR
28 23 --     qSegNeedsUnloading: HBoolArray; { Maintains a flag for each segment,
28 24 --                                     indicating whether the segment needs to
28 25 --                                     be unloaded, thereby optimizing
28 26 --                                     UnloadAllSegs. }
28 27 --
28 28 --     gCodeReserve:    Handle;    { Allocates temporary (code) reserve. }
28 29 --     gGZPurgeNotify:  ProcPtr;   { If non-NIL, then this will be called
28 30 --                                     before the Grow Zone proc purges a
28 31 --                                     handle. The proc will be passed the
28 32 --                                     actual handle. }
28 33 --
28 34 --     gMemReserve:      Handle;    { Allocates low memory reserve. }
28 35 --     gNoOfSegments:    INTEGER;   { number of code segments }
28 36 --     gOKCodeReserve:   BOOLEAN;   { if TRUE then we have an adequate code
28 37 --                                     reserve; if FALSE then the application
28 38 --                                     could crash if memory is tight }
28 39 --     gPermAllocation:  BOOLEAN;
28 40 --     gReserveExists:   BOOLEAN;   { TRUE if the code reserve is known to
28 41 --                                     be fully allocated }
28 42 --
28 43 --     { $IFC qDebug }
28 44 --     gReserveShortfall: LONGINT;  { amt. that we are lacking in the code reserve }
28 45 --     { $ENDC }
28 46 --     gSzCodeReserve:   Size;      { Attempt to reserve this much memory for
28 47 --                                     the temporary (code) reserve. }
28 48 --     gSzMemReserve:    Size;      { Attempt to reserve this much memory for
28 49 --                                     the low-memory reserve. }
28 50 --
28 51 --     gSegLoadPatch:    TrapPatch; { patch for LoadSeg }
28 52 --
28 53 -- {-----+
28 54 -- | Forward declarations |
28 55 -- +-----}
28 56 -- A FUNCTION GrowZoneProc(needed: Size): LONGINT; FORWARD;
28 57 -- A PROCEDURE BuildCodeReserve(allocLim: LONGINT; fromGZ: BOOLEAN); FORWARD;
28 58 -- A FUNCTION HandleIsEligible(h: Handle): BOOLEAN; FORWARD;
28 59 --
28 60 -- {-----+
28 61 -- | External references |
28 62 -- +-----}
28 63 -- A PROCEDURE ALoadMacAppSeg; EXTERNAL;
28 64 -- { LoadSeg is TailPatched to call ALoadMacAppSeg, which in turn calls
28 65 --   LoadMacAppSegment. ALoadMacAppSeg can only be referenced as a
28 66 --   procedure pointer, because no args are declared }
28 67 --
28 68 -- {-----+
28 69 -- | Global Routines |
28 70 -- +-----}
28 71 -- { CallNotify |
28 72 -- +-----}
28 73 -- A PROCEDURE CallNotify(h: Handle; routine: ProcPtr); INLINE $205F, $4E90;
28 74 -- {
28 75 --     MOVE.L    (A7)+,A0
28 76 --     JSR      (A0)
28 77 -- }
28 78 --
28 79 -- {-----+
28 80 -- | GZMoveHand |
28 81 -- +-----}
28 82 -- A FUNCTION GZMoveHand: Handle; INLINE $2EB8, $0330;
28 83 -- {
28 84 --     MOVE.L    $330, (A7)
28 85 -- }
28 86 --
28 87 -- {-----+
28 88 -- | GZRootHand |
28 89 -- +-----}
28 90 -- A FUNCTION GZRootHand: Handle; INLINE $2EB8, $0328;
28 91 -- {
28 92 --     MOVE.L    $328, (A7)
28 93 -- }
28 94 --
28 95 -- { $S MAINit }
28 96 --
28 97 -- {-----+
28 98 --

```



```

28 99 -- | AddAllRsrc |
28 100 -- +-----+
28 101 -- A PROCEDURE AddAllRsrc(rType: ResType; toList: HHandleList);
28 102 -- VAR
28 103 --     oldResLoad: BOOLEAN;
28 104 --     i: INTEGER;
28 105 --     h: Handle;
28 106 --     theID: INTEGER;
28 107 --     theType: ResType;
28 108 --     theName: Str255;
28 109 0- A BEGIN
28 110 --     oldResLoad := GetResLoad;
28 111 --     SetResLoad(FALSE);
28 112 --
28 113 --     FOR i := 1 TO CountResources(rType) DO
28 114 1- BEGIN
28 115 --         h := GetIndResource(rType, i);
28 116 --         GetResInfo(h, theID, theType, theName);
28 117 --
28 118 --         { If there is a ROM resource for this type and ID, don't put it
28 119 --           on the list. }
28 120 --         UserOMMap(FALSE);
28 121 --         h := GetResource(rType, theID);
28 122 --         UserOMMap(FALSE);
28 123 --         IF HomeResFile(h) <> 1 THEN
28 124 --             AddHandle(h, toList);
28 125 --
28 126 -1     END;
28 127 --
28 128 --     SetResLoad(oldResLoad);
28 129 -0 A END { AddAllRsrc };
28 130 --
28 131 --
28 132 -- { $$ MInit }
28 133 -- {-----+
28 134 -- | AddHandle |
28 135 -- +-----+
28 136 -- A PROCEDURE AddHandle(h: Handle; toList: HHandleList);
28 137 -- VAR
28 138 --     offset: LONGINT;
28 139 0- A BEGIN
28 140 --     offset := Munger(Handle(toList), 0, NIL, 0, @h, 4);
28 141 -0 A END { AddHandle };
28 142 --
28 143 --
28 144 -- { $$ MInit }
28 145 -- {-----+
28 146 -- | AddSegSizes |
28 147 -- +-----+
28 148 -- A FUNCTION AddSegSizes(segRsrc: Handle): LONGINT;
28 149 --
28 150 -- VAR
28 151 --     p: PByte;
28 152 --     oldResLoad: BOOLEAN;
28 153 --     total: LONGINT;
28 154 --     seg: Handle;
28 155 --     i: INTEGER;
28 156 --     s: Str255;
28 157 0- A BEGIN
28 158 --     MoveHHI(segRsrc);
28 159 --     HLock(segRsrc);
28 160 --
28 161 --     oldResLoad := GetResLoad;
28 162 --     SetResLoad(FALSE);
28 163 --
28 164 --     p := PByte(segRsrc^);
28 165 --     i := PInteger(p)^;
28 166 --     p := PByte(Ord(p) + 2);
28 167 --
28 168 --     total := 0;
28 169 --
28 170 --     WHILE i > 0 DO
28 171 1- BEGIN
28 172 --         BlockMove(Ptr(p), @s, p^+1);
28 173 --
28 174 --         p := PByte(Ord(p) + p^ + 1);
28 175 --         i := i - 1;
28 176 --
28 177 --         seg := GetNamedResource('CODE', s);
28 178 --         IF seg <> NIL THEN
28 179 --             total := total + SizeResource(seg) + 8;
28 180 -1         END;
28 181 --
28 182 --         AddSegSizes := total;
28 183 --
28 184 --         SetResLoad(oldResLoad);
28 185 --
28 186 --         HUnlock(segRsrc);
28 187 -0 A END { AddSegSizes };
28 188 --
28 189 --
28 190 -- { $$ MAMain }
28 191 -- { $IFC qTrace } { $D+ } { $ENDC }
28 192 -- {-----+
28 193 -- | BuildAllReserves |
28 194 -- +-----+
28 195 -- A PROCEDURE BuildAllReserves;
28 196 -- VAR
28 197 --     oldPerm: BOOLEAN;

```

```

28 198 --
28 199 0- A BEGIN
28 200 --      { set the permanent flag to ensure that the code reserve is
28 201 --      actually allocated and not given up to the low space reserve }
28 202 --      oldPerm := gPermAllocation;
28 203 --      gPermAllocation := TRUE;
28 204 --
28 205 --      { make sure code reserve is OK }
28 206 --      BuildCodeReserve(kGZMaxAlloc, FALSE);
28 207 --
28 208 --      { reallocate the low space handle, if necessary }
28 209 --      IF gMemReserve^ = NIL THEN
28 210 --          ReallocHandle(gMemReserve, gSzMemReserve);
28 211 --
28 212 --      { reset the permanent flag }
28 213 --      gPermAllocation := oldPerm;
28 214 -0 A END { BuildAllReserves };
28 215 --      {$IFC qTrace}{$D+}{$ENDC}
28 216 --
28 217 --
28 218 --      {$$ MAMain}
28 219 --      {$IFC qTrace}{$D+}{$ENDC}
28 220 --      {-----+
28 221 --      | BuildCodeReserve |
28 222 --      +-----}
28 223 -- A PROCEDURE BuildCodeReserve(allocLim: LONGINT; fromGZ: BOOLEAN);
28 224 -- VAR
28 225 --     needed:    Size;
28 226 --     avail:     Size;
28 227 --     canPurge:  Handle;
28 228 --
28 229 0- A BEGIN
28 230 --     gOKCodeReserve := TRUE; { default value }
28 231 --
28 232 --     {$IFC qDebug}
28 233 --     gReserveShortfall := 0;
28 234 --
28 235 --     IF NOT gPermAllocation THEN
28 236 --         MacsbugStr('BuildCodeReserve called with gPermAllocation = FALSE');
28 237 --     {$ENDC qDebug}
28 238 --
28 239 --     IF NOT gReserveExists THEN
28 240 1- BEGIN
28 241 --         gReserveExists := TRUE; { default value }
28 242 --
28 243 --         { free the current code reserve }
28 244 --         EmptyHandle(gCodeReserve);
28 245 --
28 246 --         { compute amt actually needed }
28 247 --         needed := Min(gSzCodeReserve - TotalTempSize(FALSE, canPurge) - 8,
28 248 --             allocLim);
28 249 --
28 250 --     IF needed > 0 THEN
28 251 2- BEGIN
28 252 --         { make as much memory available as possible }
28 253 --         IF HandleIsEligible(gMemReserve) THEN
28 254 --             EmptyHandle(gMemReserve);
28 255 --
28 256 --         IF fromGZ THEN { Never purge or compact from GrowZone }
28 257 --             avail := allocLim
28 258 --         ELSE
28 259 3- BEGIN
28 260 --             PurgeMem(needed);
28 261 --             avail := CompactMem(needed);
28 262 -3 END;
28 263 --
28 264 --         IF avail < needed THEN { could not get the whole reserve }
28 265 3- BEGIN
28 266 --             {$IFC qDebug}
28 267 --             gReserveShortfall := needed - avail;
28 268 --             {$ENDC}
28 269 --
28 270 --             gOKCodeReserve := FALSE;
28 271 --             gReserveExists := FALSE;
28 272 --
28 273 --             needed := avail; { get the most we can }
28 274 -3 END;
28 275 --
28 276 --             ReallocHandle(gCodeReserve, needed);
28 277 -2 END;
28 278 -1 END;
28 279 -0 A END { BuildCodeReserve };
28 280 --      {$IFC qTrace}{$D+}{$ENDC}
28 281 --
28 282 --
28 283 --      {$$ MAMain}
28 284 --      {-----+
28 285 --      | CheckReserve |
28 286 --      +-----}
28 287 -- A FUNCTION CheckReserve: BOOLEAN;
28 288 0- A BEGIN
28 289 --     BuildAllReserves;
28 290 --     CheckReserve := gOKCodeReserve;
28 291 -0 A END { CheckReserve };
28 292 --
28 293 --
28 294 --     {$IFC qDebug}
28 295 --     {$$ MAMain}
28 296 --     {-----+

```

```

28 297 -- | CheckRsrcUsage |
28 298 -- +-----+
28 299 -- A PROCEDURE CheckRsrcUsage;
28 300 -- VAR
28 301 --     sz: LONGINT;
28 302 --     h: Handle;
28 303 --     s: Str255;
28 304 0- A BEGIN
28 305 --     sz := TotalTempSize(TRUE, h);
28 306 --     IF sz > gMaxLockedRsrc THEN
28 307 1- BEGIN
28 308 --         gMaxLockedRsrc := sz;
28 309 --         IF gRsrcReport THEN
28 310 2- BEGIN
28 311 --             NumToString(gMaxLockedRsrc, s);
28 312 --             s := Concat(' -- New maximum resources usage: ', s, ' --');
28 313 --             ProgramReport(s, gMemMgtBreak);
28 314 -2 END;
28 315 -1 END;
28 316 -0 A END { CheckRsrcUsage };
28 317 -- {$ENDC qDebug}
28 318 --
28 319 --
28 320 -- {$IFC qDebug}
28 321 -- {$S MADEbug}
28 322 -- {-----+
28 323 -- | DoChangeReserve |
28 324 -- +-----+
28 325 -- A PROCEDURE DoChangeReserve(alter: BOOLEAN;
28 326 --                               VAR codeReserve, codeShort, lowSpaceReserve: LONGINT;
28 327 --                               VAR gotCode, gotLowSpace: BOOLEAN);
28 328 --
28 329 -- VAR
28 330 --     x: LONGINT;
28 331 --     s: Str255;
28 332 --
28 333 0- A BEGIN
28 334 --     IF alter THEN
28 335 1- BEGIN
28 336 --         Write('code reserve size = ', gSzCodeReserve:1, ' ');
28 337 --         IF gOKCodeReserve THEN
28 338 --             Writeln(' (OK)')
28 339 --         ELSE
28 340 --             Writeln(' (gone)');
28 341 --
28 342 --         Write('low space reserve size = ', gSzMemReserve:1, ' ');
28 343 --         IF gMemReserve^ <> NIL THEN
28 344 --             Writeln(' (OK)')
28 345 --         ELSE
28 346 --             Writeln(' (gone)');
28 347 --
28 348 --         Writeln;
28 349 --
28 350 --         Write('New code reserve (-1 = no change): ');
28 351 --         Readln(x);
28 352 --         IF x >= 0 THEN
28 353 --             codeReserve := x
28 354 --         ELSE
28 355 --             codeReserve := gSzCodeReserve;
28 356 --
28 357 --         Write('New low space reserve (-1 = no change): ');
28 358 --         Readln(x);
28 359 --         IF x >= 0 THEN
28 360 --             lowSpaceReserve := x
28 361 --         ELSE
28 362 --             lowSpaceReserve := gSzMemReserve;
28 363 --
28 364 --         Write('Reset max resource usage (Y or N) [N]? ');
28 365 --         Readln(s);
28 366 --         IF s <> '' THEN
28 367 --             IF (s[1] = 'y') OR (s[1] = 'Y') THEN
28 368 2- BEGIN
28 369 --                 gMaxLockedRsrc := 0;
28 370 -2 END;
28 371 --
28 372 --         Writeln;
28 373 --
28 374 --         SetReserveSize(codeReserve, lowSpaceReserve);
28 375 -1 END
28 376 --     ELSE
28 377 --         BuildAllReserves;
28 378 --
28 379 --         codeReserve := gSzCodeReserve;
28 380 --         codeShort := gReserveShortfall;
28 381 --         lowSpaceReserve := gSzMemReserve;
28 382 --         gotCode := gOKCodeReserve;
28 383 --         gotLowSpace := gMemReserve^ <> NIL;
28 384 -0 A END { DoChangeReserve };
28 385 -- {$ENDC qDebug}
28 386 --
28 387 --
28 388 -- {$IFC qTrace}{$D+}{$ENDC}
28 389 -- {$S MAINit}
28 390 -- {-----+
28 391 -- | DoInitUMemory |
28 392 -- +-----+
28 393 -- A PROCEDURE DoInitUMemory (VAR sizeTempReserve, sizeLowSpaceReserve: Size);
28 394 --
28 395 -- { Called from InitUMemory so that InitUMemory can be in the main segment

```

```

28 396 --          and this code can be in another segment. )
28 397 --
28 398 --      TYPE
28 399 --          pINTEGER = ^INTEGER;
28 400 --          hMem = ^pMem;
28 401 --          pMem = ^Mem;
28 402 --          Mem = RECORD
28 403 --              codeVal,
28 404 --              lowSpaceVal,
28 405 --              stackVal: LONGINT;
28 406 --          END;
28 407 --
28 408 --      VAR
28 409 --          i: INTEGER;
28 410 --          oldResLoad: BOOLEAN;
28 411 --          seg: Handle;
28 412 --          aZone: THz;
28 413 --          StackTot: LONGINT;
28 414 --          h: Handle;
28 415 --
28 416 0- A BEGIN
28 417 --      { Initialize memory management globals }
28 418 --      gPermAllocation := FALSE;
28 419 --      gMemReserve := NewHandle(0);
28 420 --      gSzMemReserve := 0;
28 421 --      gCodeReserve := NewHandle(0);
28 422 --      gSzCodeReserve := 0;
28 423 --      gGZPurgeNotify := NIL;
28 424 --      gOKCodeReserve := TRUE;
28 425 --      gReserveExists := FALSE;
28 426 --      {$IFC qDebug}
28 427 --      gSegReport := FALSE;
28 428 --      gUnloadAllSegs := TRUE;
28 429 --      {$ENDC}
28 430 --
28 431 --      aZone := ApplicZone;
28 432 --      aZone^.flags := BOR(aZone^.flags, $0400);
28 433 --      { set the Memory Manager bit that says to always call the
28 434 --      Grow Zone proc, even in "non-critical" situations }
28 435 --
28 436 --      SetGrowZone(@GrowZoneProc);
28 437 --
28 438 --      { Create a list of code segment handles. This assumes that (1) code
28 439 --      segments have consecutive ids starting with 1 }
28 440 --
28 441 --      i := CountResources('CODE') - 1; { Can't use Count1Resources since this
28 442 --      must work on 64K ROMs. }
28 443 --
28 444 --      gCodeSegs := HHandleList(NewHandle(i * SIZEOF(Handle)));
28 445 --
28 446 --      gNoOfSegments := 0;
28 447 --      oldResLoad := GetResLoad;
28 448 --      SetResLoad(FALSE);
28 449 --      WHILE i > 0 DO
28 450 1- BEGIN
28 451 --          seg := GetResource('CODE', i);
28 452 --          gCodeSegs^[i] := seg;
28 453 --          IF seg <> NIL THEN
28 454 --              gNoOfSegments := gNoOfSegments + 1;
28 455 --              i := i - 1;
28 456 -1 END;
28 457 --      SetResLoad(oldResLoad);
28 458 --
28 459 --      SetHandlesSize(Handle(gCodeSegs), SIZEOF(Handle) * gNoOfSegments);
28 460 --
28 461 --      gNonResidents := gCodeSegs;
28 462 --      IF HandToHand(Handle(gNonResidents)) <> noErr THEN { ??? }
28 463 --
28 464 --      gSegNeedsUnloading := HBoolArray(NewHandle(SIZEOF(BOOLEAN) * gNoOfSegments));
28 465 --      FOR i := 1 TO gNoOfSegments DO
28 466 --          gSegNeedsUnloading^[i] := TRUE;
28 467 --
28 468 --      gNonResidents^[1] := NIL; { Main is always non resident }
28 469 --      gSegNeedsUnloading^[1] := FALSE;
28 470 --
28 471 --      { init the gSysMemList }
28 472 --      gSysMemList := HHandleList(NewHandle(0));
28 473 --      AddAllRsrc('LDEF', gSysMemList);
28 474 --      AddAllRsrc('CDEF', gSysMemList);
28 475 --      AddAllRsrc('MDEF', gSysMemList);
28 476 --      AddAllRsrc('WDEF', gSysMemList);
28 477 --      AddAllRsrc('PACK', gSysMemList);
28 478 --
28 479 --      { Compute memory slop needed }
28 480 --      sizeTempReserve := 0;
28 481 --      sizeLowSpaceReserve := 0;
28 482 --      stackTot := 0;
28 483 --
28 484 --      FOR i := 1 TO CountResources('seg!') DO
28 485 1- BEGIN
28 486 --          h := GetIndResource('seg!', i);
28 487 --          sizeTempReserve := sizeTempReserve + AddSegSizes(h);
28 488 --          ReleaseResource(h);
28 489 -1 END;
28 490 --
28 491 --      FOR i := 1 TO CountResources('mem!') DO
28 492 1- BEGIN
28 493 --          h := GetIndResource('mem!', i);
28 494 --          WITH hMem(h)^^ DO

```

```

28 495 2-      BEGIN
28 496 --      sizeTempReserve := sizeTempReserve + codeVal;
28 497 --      sizeLowSpaceReserve := sizeLowSpaceReserve + lowSpaceVal;
28 498 --      stackTot := stackTot + stackVal;
28 499 -2      END;
28 500 --      ReleaseResource(h);
28 501 -1      END;
28 502 --
28 503 --      SetStackSpace(stackTot);
28 504 --
28 505 --      MaxApplZone;
28 506 --
28 507 --      gApplMemList := NIL;
28 508 --      gApp2MemList := NIL;
28 509 --
28 510 --      { Set up the LoadSeg patch }
28 511 --
28 512 --      {$IFC (NOT qNeedsROM128K) OR qDebug}
28 513 --      {$IFC NOT qDebug}      { If debugging, patch the trap anyway}
28 514 --      IF NOT gConfiguration.hasROM128K THEN
28 515 --      {$ENDC}
28 516 --      FailOSErr(PatchTrap(gSegLoadPatch, $A9F0, @ALoadMacAppSeg));  { LoadSeg }
28 517 --      {$ENDC}
28 518 --
28 519 -0 A  END { DoInitUMemory };
28 520 --      {$IFC qTrace}{$D+}{$ENDC}
28 521 --
28 522 --
28 523 --      {$S MAMain}
28 524 --      {-----+
28 525 --      | FailNoReserve |
28 526 --      +-----}
28 527 -- A  PROCEDURE FailNoReserve;
28 528 0- A  BEGIN
28 529 --      IF NOT CheckReserve THEN
28 530 --      Failure(memFullErr, 0);
28 531 -0 A  END { FailNoReserve };
28 532 --
28 533 --
28 534 --      {$S MAMain}
28 535 --      {-----+
28 536 --      | FailSpaceIsLow |
28 537 --      +-----}
28 538 -- A  PROCEDURE FailSpaceIsLow;
28 539 0- A  BEGIN
28 540 --      IF MemSpaceIsLow THEN
28 541 --      Failure(memFullErr, 0);
28 542 -0 A  END { FailSpaceIsLow };
28 543 --
28 544 --
28 545 --      {$IFC qTrace}{$D+}{$ENDC}
28 546 --      {$S MAMain}
28 547 --      {-----+
28 548 --      | GetReserveSize |
28 549 --      +-----}
28 550 -- A  PROCEDURE GetReserveSize (VAR szCodeReserve, szMemReserve: SIZE);
28 551 --
28 552 0- A  BEGIN
28 553 --      szCodeReserve := gSzCodeReserve;
28 554 --      szMemReserve := gSzMemReserve;
28 555 -0 A  END { GetMemReserveSize };
28 556 --      {$IFC qTrace}{$D+}{$ENDC}
28 557 --
28 558 --
28 559 --      {$IFC qTrace}{$D+}{$ENDC} { no %BP/%EP allowed in here, because we cannot call to any
28 560 --      other segment from this procedure }
28 561 --      {$S Main} { must be in Main segment because we call this in order to make the resident segment resident }
28 562 --      {-----+
28 563 --      | GetSegNumber |
28 564 --      +-----}
28 565 -- A  FUNCTION GetSegNumber(aProc: ProcPtr): INTEGER;
28 566 --      TYPE
28 567 --      PInteger = ^INTEGER;
28 568 --
28 569 --      VAR
28 570 --      i:      INTEGER;
28 571 --      jt:     LONGINT;
28 572 --      segNum: INTEGER;
28 573 --      seg:    Handle;
28 574 --      segStart: LONGINT;
28 575 --
28 576 0- A  BEGIN
28 577 --      IF PInteger(aProc)^ = $4EF9 THEN { loaded segment }
28 578 --      GetSegNumber := PInteger(ORD(aProc)-2)^
28 579 --      ELSE IF PInteger(aProc)^ = $3F3C THEN { unloaded segment }
28 580 --      GetSegNumber := PInteger(ORD(aProc)+2)^
28 581 --      ELSE { routine that computed @proc was in same segment as the proc }
28 582 1-      BEGIN
28 583 --      {$IFC qDebug}
28 584 --      ProgramBreak('GetSegNumber was not passed an jump table address');
28 585 --      {$ENDC}
28 586 --      GetSegNumber := 0;
28 587 -1      END;
28 588 -0 A  END { GetSegNumber };
28 589 --      {$IFC qTrace}{$D+}{$ENDC}
28 590 --
28 591 --
28 592 --      {$S MAMain}
28 593 --      {$IFC qTrace}{$D+}{$ENDC}

```

```

28 594 -- {-----+
28 595 -- |   GetSegSize
28 596 -- +-----}
28 597 -- A FUNCTION GetSegSize(segnum: INTEGER): Size;
28 598 --
28 599 -- VAR
28 600 --     curResLoad:    BOOLEAN;
28 601 0- A BEGIN
28 602 --     curResLoad := GetResLoad;
28 603 --     SetResLoad(FALSE);
28 604 --     GetSegSize := SizeResource(GetResource('CODE', segnum));
28 605 --     SetResLoad(curResLoad);
28 606 0- A END { GetSegSize };
28 607 --     { $IFC qTrace } { $D+ } { $ENDC }
28 608 --
28 609 --
28 610 --     { $S Main }
28 611 --     { $IFC qTrace } { $D+ } { $ENDC }
28 612 -- {-----+
28 613 -- |   GrowZoneProc
28 614 -- +-----}
28 615 -- A FUNCTION GrowZoneProc(needed: Size): LONGINT;
28 616 -- VAR
28 617 --     result:        LONGINT;
28 618 --     canPurge:      Handle;
28 619 --     codeSize:      Size;
28 620 --     reserveSize:   LONGINT;
28 621 --
28 622 0- A BEGIN
28 623 --     SetupA5;
28 624 --
28 625 --     result := 0; { default is to fail }
28 626 --
28 627 --     { on a temp alloc, free all code slack immediately }
28 628 --     IF NOT gPermAllocation THEN
28 629 --         IF HandleIsEligible(gCodeReserve) THEN
28 630 1-             BEGIN
28 631 --                 EmptyHandle(gCodeReserve);
28 632 --                 gReserveExists := FALSE;
28 633 --                 result := 1;
28 634 1-             END;
28 635 --
28 636 --     IF result = 0 THEN { try harder: see if we can purge a code segment
28 637 --                         or reduce the code reserve handle }
28 638 1-         BEGIN
28 639 --             { compute size of resources currently in memory }
28 640 --
28 641 --             codeSize := TotalTempSize(FALSE, canPurge);
28 642 --
28 643 --             { see if the code reserve handle is too large }
28 644 --
28 645 --             IF HandleIsEligible(gCodeReserve) THEN
28 646 --                 { we have a code reserve handle; this implies permanent allocation,
28 647 --                 otherwise the handle would have been emptied above }
28 648 2-             BEGIN
28 649 --                 reserveSize := GetHandleSize(gCodeReserve);
28 650 --
28 651 --                 { the following test is an optimization to avoid calling
28 652 --                 BuildCodeReserve if there is no hope of reducing
28 653 --                 the code reserve handle }
28 654 --                 IF codeSize + reserveSize + 8 > gSzCodeReserve THEN
28 655 3-                     BEGIN { reserve is too big }
28 656 --                         gReserveExists := FALSE;
28 657 --                         { this should lower the code reserve }
28 658 --                         BuildCodeReserve(reserveSize, TRUE);
28 659 --
28 660 --                         { see if we succeeded in freeing some memory }
28 661 --                         IF gCodeReserve^ = NIL THEN
28 662 --                             result := 1
28 663 --                         ELSE IF GetHandleSize(gCodeReserve) < reserveSize THEN
28 664 --                             result := 1;
28 665 3-                     END;
28 666 2-             END;
28 667 --
28 668 --     IF result = 0 THEN
28 669 --         IF canPurge <> NIL THEN
28 670 --             { got something; only purge it if this is temporary
28 671 --             OR we know there is too much code in memory already }
28 672 --             IF NOT gPermAllocation OR (gCodeReserve^ = NIL) THEN
28 673 2-                 BEGIN
28 674 --                     IF gGZPurgeNotify <> NIL THEN
28 675 --                         CallNotify(canPurge, gGZPurgeNotify);
28 676 --
28 677 --                     reserveSize := GetHandleSize(canPurge);
28 678 --                     HPurge(canPurge);
28 679 --                     EmptyHandle(canPurge);
28 680 --                     gReserveExists := FALSE;
28 681 --
28 682 --                     IF gPermAllocation THEN { don't free too much however }
28 683 --                         BuildCodeReserve(reserveSize, TRUE);
28 684 --
28 685 --                     result := 1;
28 686 2-                 END;
28 687 1-             END;
28 688 --
28 689 --     IF result = 0 THEN { last ditch attempt: free emergency reserve }
28 690 --         IF HandleIsEligible(gMemReserve) THEN
28 691 1-             BEGIN
28 692 --                 EmptyHandle(gMemReserve);

```

```

28 693 --          result := 1;
28 694 --          END;
28 695 --
28 696 --          GrowZoneProc := result;
28 697 --
28 698 --          RestoreA5;
28 699 -- A      END { GrowZoneProc };
28 700 --          { $IFC qTrace } { $D++ } { $ENDC }
28 701 --
28 702 --
28 703 --          { $S Main }
28 704 --          { $IFC qTrace } { $D++ } { $ENDC }
28 705 --          { -----+
28 706 --          |   HandleIsEligible
28 707 --          +----- }
28 708 -- A      FUNCTION HandleIsEligible(h: Handle): BOOLEAN;
28 709 -- A      BEGIN
28 710 --          IF h = NIL THEN
28 711 --              HandleIsEligible := FALSE
28 712 --          ELSE
28 713 --              HandleIsEligible := (h <> GZMoveHand) AND (h <> GZRootHand);
28 714 -- A      END { HandleIsEligible };
28 715 --          { $IFC qTrace } { $D++ } { $ENDC }
28 716 --
28 717 --
28 718 --          { $IFC qTrace } { $D++ } { $ENDC }
28 719 --          { $S Main }          { Must be in main segment and called from main segment }
28 720 --          { -----+
28 721 --          |   InitUMemory
28 722 --          +----- }
28 723 -- A      FUNCTION InitUMemory: BOOLEAN;
28 724 --
28 725 --          VAR
28 726 --              codeRes,
28 727 --              lowSpaceRes:   Size;
28 728 --
28 729 -- A      BEGIN
28 730 --          DoInitUMemory(codeRes, lowSpaceRes);
28 731 --          UnloadAllSegments; { get init segment(s) out of middle of heap, so
28 732 --                               SetReserveSize has maximum space to work with }
28 733 --
28 734 --          SetReserveSize(codeRes, lowSpaceRes);
28 735 --          InitUMemory := gOKCodeReserve;
28 736 -- A      END { InitUMemory };
28 737 --          { $IFC qTrace } { $D++ } { $ENDC }
28 738 --
28 739 --
28 740 --          { $IFC qTrace } { $D++ } { $ENDC } { no _BP/_EP allowed in here, because we cannot call to any
28 741 --                                               other segment from this procedure }
28 742 --          { $S Main } { must be in Main segment }
28 743 --          { -----+
28 744 --          |   LoadMacAppSegment
28 745 --          +----- }
28 746 -- A      FUNCTION LoadMacAppSegment(segNum: INTEGER): LONGINT;
28 747 --
28 748 --          { This function patches LoadSeg. It is called from the assembly-language
28 749 --            routine AMacAppLoadSeg, which is the actual patch setup by PatchTrap.
28 750 --            AMacAppLoadSeg saves D0-D1-A0/A1 and sets up the stack by
28 751 --            1) allocating space for LoadMacAppSegment's result, and
28 752 --            2) pushing a copy of LoadSeg's parameter onto the stack for use
28 753 --            by LoadMacAppSegment. }
28 754 --
28 755 --          { $IFC qDebug }
28 756 --          VAR
28 757 --              id:           INTEGER;
28 758 --              kind:         ResType;
28 759 --              segName:      Str255;
28 760 --          { $ENDC }
28 761 --
28 762 -- A      BEGIN
28 763 --          SetupA5;          { ***** Called from trap patches ***** }
28 764 --
28 765 --          LoadMacAppSegment := gSegLoadPatch.oldTrapAddr;
28 766 --
28 767 --          IF PreloadSegment(segNum) THEN { ??? what to do if fails ??? }
28 768 --
28 769 --              gSegNeedsUnloading^[segNum] := TRUE;
28 770 --
28 771 --          { $IFC qDebug }
28 772 --          IF gSegReport THEN
28 773 --              BEGIN
28 774 --                  { Cause the debugger to break at the start of the next routine. }
28 775 --                  gReportNext := TRUE;
28 776 --                  GetResInfo(gCodeSegs^[segNum], id, kind, segName);
28 777 --                  NumToString(segNum, gReportInfo);
28 778 --                  gReportInfo := CONCAT(' *** Segment Loaded: ', gReportInfo, ' ', segName);
28 779 --                  gSingleStep := gMemMgtBreak;
28 780 --              END;
28 781 --          { $ENDC }
28 782 --
28 783 --          RestoreA5;
28 784 --
28 785 -- A      END { LoadMacAppSegment };
28 786 --          { $IFC qTrace } { $D++ } { $ENDC }
28 787 --
28 788 --
28 789 --          { $S Main }          { Must be in Main segment }
28 790 --          { -----+
28 791 --          |   LoadResidentSegments

```

```

28 792 -- +-----}
28 793 -- A PROCEDURE LoadResidentSegments;
28 794 --
28 795 -- VAR
28 796 --     resIndex:    INTEGER;
28 797 --     i:           INTEGER;
28 798 --     offset:      INTEGER;
28 799 --     nameList:    Handle;
28 800 --     segNumber:   INTEGER;
28 801 --     p:           PByte;
28 802 --     name:        Str255;
28 803 --     seg:         Handle;
28 804 --     theType:     ResType;
28 805 --
28 806 0- A BEGIN
28 807 --     FOR resIndex := 1 TO CountResources('res!') DO
28 808 1-         BEGIN
28 809 --             nameList := GetIndResource('res!', resIndex);
28 810 --             HNoPurge(nameList);
28 811 --
28 812 --             offset := 2;
28 813 --             FOR i := 1 TO PInteger(nameList)^ DO
28 814 2-                 BEGIN
28 815 --                     p := PByte(ORD4(nameList^) + offset);
28 816 --                     BlockMove(Ptr(p), @name, p^+1);
28 817 --                     offset := offset + LENGTH(name) + 1;
28 818 --
28 819 --                     seg := GetNamedResource('CODE', name);
28 820 --                     IF seg <> NIL THEN
28 821 3-                         BEGIN
28 822 --                             GetResInfo(seg, segNumber, theType, name);
28 823 --                             SetResidentSegment(segNumber, TRUE);
28 824 -3                             END;
28 825 -2                         END;
28 826 --
28 827 --                     HPurge(nameList);
28 828 --                     ReleaseResource(nameList);
28 829 -1                     END;
28 830 -0 A END {LoadResidentSegments};
28 831 --
28 832 --
28 833 -- { $$ MAMain }
28 834 -- {-----+
28 835 -- | MemSpaceIsLow |
28 836 -- +-----}
28 837 -- A FUNCTION MemSpaceIsLow: BOOLEAN;
28 838 0- A BEGIN
28 839 --     BuildAllReserves;
28 840 --
28 841 --     MemSpaceIsLow := gMemReserve^ = NIL;
28 842 -0 A END { MemSpaceIsLow };
28 843 --
28 844 --
28 845 -- { $IFC qTrace } { $D+ } { $ENDC }
28 846 -- { $$ MAMain }
28 847 -- {-----+
28 848 -- | NewPermHandle |
28 849 -- +-----}
28 850 -- A FUNCTION NewPermHandle(logicalSize: Size): Handle;
28 851 -- VAR
28 852 --     priorPerm: BOOLEAN;
28 853 --
28 854 0- A BEGIN
28 855 --     priorPerm := PermAllocation(TRUE);
28 856 --     NewPermHandle := NewHandle(logicalSize);
28 857 --     priorPerm := PermAllocation(priorPerm);
28 858 -0 A END { NewPermHandle };
28 859 -- { $IFC qTrace } { $D+ } { $ENDC }
28 860 --
28 861 --
28 862 -- { $IFC qTrace } { $D+ } { $ENDC }
28 863 -- { $$ MAMain }
28 864 -- {-----+
28 865 -- | PermAllocation |
28 866 -- +-----}
28 867 -- A FUNCTION PermAllocation(permanent: BOOLEAN): BOOLEAN;
28 868 -- VAR
28 869 --     b: BOOLEAN;
28 870 --
28 871 0- A BEGIN
28 872 --     PermAllocation := gPermAllocation;
28 873 --
28 874 --     IF permanent <> gPermAllocation THEN
28 875 1-         BEGIN
28 876 --             gPermAllocation := permanent;
28 877 --
28 878 --             IF permanent THEN
28 879 --                 BuildCodeReserve(kGZMaxAlloc, FALSE);
28 880 -1             END;
28 881 -0 A END { PermAllocation };
28 882 -- { $IFC qTrace } { $D+ } { $ENDC }
28 883 --
28 884 --
28 885 -- { $IFC qTrace } { $D+ } { $ENDC } { no %_BP/%_EP allowed in here, because we cannot call to any
28 886 --                                     other segment from this procedure }
28 887 -- { $$ Main } { must be in Main segment }
28 888 -- {-----+
28 889 -- | PreloadSegment |
28 890 -- +-----}

```



```

28 891 -- A FUNCTION PreloadSegment(segNum: INTEGER): BOOLEAN;
28 892 -- TYPE
28 893 --     HSegHeader = ^PSegHeader;
28 894 --     PSegHeader = ^SegHeader;
28 895 --     SegHeader = RECORD
28 896 --         jtOffset: INTEGER;
28 897 --         jtCount: INTEGER;
28 898 --     END;
28 899 --     PInteger = ^INTEGER;
28 900 --
28 901 -- VAR
28 902 --     seg: Handle;
28 903 --     err: OSErr;
28 904 0- A BEGIN
28 905 --     {$IFC qDebug}
28 906 --         IF gPermAllocation THEN
28 907 1-             BEGIN
28 908 --                 Writeln('segment # = ', segNum:1);
28 909 --                 ProgramBreak('Trying to load a segment with PermAllocation = TRUE. ');
28 910 -1             END;
28 911 --     {$ENDC}
28 912 --
28 913 --     seg := GetResource('CODE', segNum);
28 914 --
28 915 --     IF seg = NIL THEN
28 916 --         PreloadSegment := FALSE
28 917 --     ELSE
28 918 1-         BEGIN
28 919 --             PreloadSegment := TRUE;
28 920 --
28 921 --             IF NOT BTst(GetHandleBits(seg), 7) THEN { not yet locked }
28 922 2-                 BEGIN
28 923 --                     MoveHHI(seg); { ??? check MemErr ??? }
28 924 --                     HLock(seg);
28 925 -2                     END;
28 926 -1                 END;
28 927 -0 A     END { PreloadSegment };
28 928 --     {$IFC qTrace}{$D+}{$ENDC}
28 929 --
28 930 --
28 931 --     {$S MAMain}
28 932 --     {-----+
28 933 --     | RemHandle |
28 934 --     +-----+}
28 935 -- A PROCEDURE RemHandle(h: Handle; toList: HHandleList);
28 936 -- VAR
28 937 --     offset: LONGINT;
28 938 0- A BEGIN
28 939 --     offset := Munger(Handle(toList), 0, @h, SIZEOF(Handle), @h, 0);
28 940 -0 A END { RemHandle };
28 941 --
28 942 --
28 943 --     {$S MAMain}
28 944 --     {$IFC qTrace}{$D+}{$ENDC}
28 945 --     {-----+
28 946 --     | ScanHandles |
28 947 --     +-----+}
28 948 -- A PROCEDURE ScanHandles(PROCEDURE DoToHandle(h: Handle));
28 949 --
28 950 --     { Calls DoToHandle for each handle in gCodeSegs, gSysMemList,
28 951 --       gApplMemList and gApp2MemList. }
28 952 --
28 953 --     {-----+
28 954 --     | ScanList |
28 955 --     +-----+}
28 956 -- B PROCEDURE ScanList(list: HHandleList);
28 957 -- TYPE
28 958 --     PtrHandle = ^Handle;
28 959 --
28 960 -- VAR
28 961 --     i: INTEGER;
28 962 --     p: PtrHandle;
28 963 0- B BEGIN
28 964 --     i := GetHandleSize(Handle(list)) DIV SIZEOF(Handle);
28 965 --
28 966 --     p := PtrHandle(list^);
28 967 --     WHILE i > 0 DO
28 968 1-         BEGIN
28 969 --             DoToHandle(p^);
28 970 --             p := PtrHandle(Ord(p) + SIZEOF(Handle));
28 971 --             i := i - 1;
28 972 -1         END;
28 973 -0 B     END { ScanList };
28 974 --
28 975 0- A BEGIN
28 976 --     ScanList(gCodeSegs);
28 977 --     IF gApplMemList <> NIL THEN
28 978 --         ScanList(gApplMemList);
28 979 --     ScanList(gSysMemList);
28 980 --     IF gApp2MemList <> NIL THEN
28 981 --         ScanList(gApp2MemList);
28 982 -0 A     END { ScanHandles };
28 983 --     {$IFC qTrace}{$D+}{$ENDC}
28 984 --
28 985 --
28 986 --     {$IFC qTrace}{$D+}{$ENDC}
28 987 --     {$S MAMain}
28 988 --     {-----+
28 989 --     | SetReserveSize |

```

```

28 990 -- +-----}
28 991 -- A PROCEDURE SetReserveSize(forCode, forOther: Size);
28 992 -- VAR
28 993 --     oldPerm:    BOOLEAN;
28 994 0- A BEGIN
28 995 --     gSzCodeReserve := forCode;
28 996 --     gSzMemReserve := forOther;
28 997 --
28 998 --     { Since the numbers have changed, make sure we start from scratch. }
28 999 --     gReserveExists := FALSE;
28 1000 --     EmptyHandle(gMemReserve);
28 1001 --
28 1002 --     BuildAllReserves;
28 1003 0- A END { SetReserveSize };
28 1004 --     { $IFC qTrace } { $D++ } { $ENDC }
28 1005 --
28 1006 --
28 1007 --     { $IFC qTrace } { $D++ } { $ENDC } { no %_BP/%_EP allowed in here, because we cannot call to any
28 1008 --                                     other segment from this procedure }
28 1009 --     { $$ Main } { must be in Main segment }
28 1010 --     {-----}
28 1011 --     | SetResidentSegment |
28 1012 --     +-----}
28 1013 -- A PROCEDURE SetResidentSegment(segNum: INTEGER; makeResident: BOOLEAN);
28 1014 0- A BEGIN
28 1015 --     IF segNum > 1 THEN { can't modify the blank segment (# 1) }
28 1016 --         IF makeResident THEN
28 1017 1- BEGIN
28 1018 --             gNonResidents^^[segNum] := NIL;
28 1019 --             gSegNeedsUnloading^^[segNum] := FALSE;
28 1020 --             IF PreloadSegment(segNum) THEN ; { ??? what if this fails ??? }
28 1021 1- END
28 1022 --         ELSE
28 1023 1- BEGIN
28 1024 --             gNonResidents^^[segNum] := gCodeSegs^^[segNum];
28 1025 --             gSegNeedsUnloading^^[segNum] := TRUE;
28 1026 1- END;
28 1027 0- A END { SetResidentSegment };
28 1028 --     { $IFC qTrace } { $D++ } { $ENDC }
28 1029 --
28 1030 --
28 1031 --     { $$ MAMain }
28 1032 --     { $IFC qTrace } { $D++ } { $ENDC }
28 1033 --     {-----}
28 1034 --     | SetStackSpace |
28 1035 --     +-----}
28 1036 -- A PROCEDURE SetStackSpace(numBytes: LONGINT);
28 1037 --
28 1038 --     CONST
28 1039 --         CurStackBase = $0908;
28 1040 --
28 1041 --     VAR
28 1042 --         curLimit: LONGINT;
28 1043 --         newLimit: LONGINT;
28 1044 --
28 1045 0- A BEGIN
28 1046 --         newLimit := PLongInt(CurStackBase) - numBytes;
28 1047 --
28 1048 --         IF Ord(GetApplLimit) > newLimit THEN
28 1049 --             SetApplLimit(Ptr(newLimit));
28 1050 0- A END { SetStackSpace };
28 1051 --     { $IFC qTrace } { $D++ } { $ENDC }
28 1052 --
28 1053 --
28 1054 --     { $$ MAMain }
28 1055 --     { $IFC qTrace } { $D++ } { $ENDC }
28 1056 --     {-----}
28 1057 --     | TotalTempSize |
28 1058 --     +-----}
28 1059 -- A FUNCTION TotalTempSize(justLocked: BOOLEAN; VAR canPurge: Handle): Size;
28 1060 -- VAR
28 1061 --     total:    Size;
28 1062 --     applZone: THz;
28 1063 --
28 1064 --     {-----}
28 1065 --     | TotalUp |
28 1066 --     +-----}
28 1067 -- B PROCEDURE TotalUp(h: Handle);
28 1068 -- VAR
28 1069 --     hIsLocked: BOOLEAN;
28 1070 0- B BEGIN
28 1071 --     IF h <> NIL THEN { in memory already }
28 1072 --         IF HandleZone(h) = applZone THEN { in application heap }
28 1073 1- BEGIN
28 1074 --             HNoPurge(h);
28 1075 --
28 1076 --             hIsLocked := BTest(GetHandleBits(h), 7);
28 1077 --
28 1078 --             IF NOT justLocked OR hIsLocked THEN
28 1079 --                 total := total + GetHandleSize(h) + 8;
28 1080 --                 { add in the size plus heap overhead }
28 1081 --
28 1082 --             IF (NOT hIsLocked) THEN
28 1083 --                 IF canPurge = NIL THEN
28 1084 --                     IF HandleIsEligible(h) THEN
28 1085 --                         canPurge := h;
28 1086 1- END;
28 1087 0- B END { TotalUp };
28 1088 --

```

```

28 1089 0- A BEGIN
28 1090 --      canPurge := NIL;
28 1091 --      total := 0;
28 1092 --      applZone := ApplicZone;
28 1093 --
28 1094 --      ScanHandles(TotalUp);
28 1095 --
28 1096 --      TotalTempSize := total;
28 1097 -0 A END { TotalTempSize };
28 1098 --      { $IFC qTrace } { $D++ } { $ENDC }
28 1099 --
28 1100 --
28 1101 --      { $IFC qTrace } { $D++ } { $ENDC } { no %_BP/%_EP allowed in here, because we cannot call to any
28 1102 --                                     other segment from this procedure }
28 1103 --      { $$ Main } { must be in Main segment }
28 1104 --      {-----+
28 1105 --      |      UnloadAllSegments
28 1106 --      +-----}
28 1107 -- A PROCEDURE UnloadAllSegments;
28 1108 --
28 1109 --      CONST
28 1110 --          CurJTOffset =    $0934;
28 1111 --
28 1112 --      VAR
28 1113 --          i:                LONGINT;
28 1114 --          seg:              Handle;
28 1115 --          base:             LONGINT;
28 1116 --          oldResLoad:       BOOLEAN;
28 1117 --
28 1118 0- A BEGIN
28 1119 --      { $IFC qDebug }
28 1120 --          CheckRsrcUsage;
28 1121 --      { $ENDC }
28 1122 --
28 1123 --      { $IFC qDebug }
28 1124 --          IF gUnloadAllSegs THEN
28 1125 --      { $ENDC }
28 1126 1-          BEGIN
28 1127 --              { $%+ }
28 1128 --              base := %_GetA5 + PInteger(CurJTOffset)^;
28 1129 --              { $%- }
28 1130 --
28 1131 --              FOR i := 2 TO gNoOfSegments DO      { 1st segment is never unloaded }
28 1132 --                  IF gSegNeedsUnloading^[i] THEN
28 1133 2-                      BEGIN
28 1134 --                          seg := gNonResidents^[i];
28 1135 --                          IF seg <> NIL THEN
28 1136 --                              IF seg^ <> NIL THEN
28 1137 --                                  UnloadSeg(Ptr(base + HInteger(seg)^ + 2));
28 1138 --                                  gSegNeedsUnloading^[i] := FALSE;
28 1139 -2                      END;
28 1140 --
28 1141 --      { $IFC qDebug }
28 1142 --          IF gSegReport THEN
28 1143 --              ProgramReport(' *** Just unloaded all segments ****, gMemMgtBreak);
28 1144 --      { $ENDC }
28 1145 -1          END;
28 1146 -0 A END { UnloadAllSegments };
28 1147 --      { $IFC qTrace } { $D++ } { $ENDC }
28 1148 --
28 1149 --
28 1150 --      { $IFC qDebug }
28 1151 --      { $$ MAINinspect }
28 1152 --      {-----+
28 1153 --      |      WriteReserves
28 1154 --      +-----}
28 1155 -- A PROCEDURE WriteReserves;
28 1156 --
28 1157 0- A BEGIN
28 1158 --      WrLblPtr('Temporary reserve (gCodeReserve)', gCodeReserve); Writeln;
28 1159 --      WrLblPtr('Low-memory reserve (gMemReserve)', gMemReserve); Writeln;
28 1160 -0 A END { WriteReserves };
28 1161 --      { $ENDC }
27 368 --
27 369 --      END { UMemory }.

```

```

29 1 -- { $P }
29 2 -- { UMenuSetup.p }
29 3 -- { Copyright © 1984-1988 Apple Computer Inc. All rights reserved. }
29 4 --
29 5 -- { Conditional compile flags used by this unit:
29 6 --
29 7 --     qDebug      True includes debugging code (writeln and ProgramBreak),
29 8 --                false ignores this code.
29 9 --
29 10 --     qTrace      True surrounds trap patches with $D+ and $D++ (because
29 11 --                the default is $D++), false skips these directives.
29 12 --
29 13 --     The settings of $D, $R and $N are imported from the UMAUtil interface.
29 14 -- }
29 15 --
29 16 --
29 17 -- UNIT UMenuSetup;
29 18 --
29 19 -- {
29 20 --
29 21 --     T H E O R Y   O F   O P E R A T I O N
29 22 --
29 23 --     This unit provides two features: It implements a command numbering
29 24 --     system that is independent of menu/item placement, and implements a
29 25 --     framework for optimizing menu setups.
29 26 --
29 27 --     The command numbering system works by assigning commands a unique
29 28 --     integer number, and providing a mechanism for mapping command numbers
29 29 --     to menu/item pairs. A set of routines are provide to manipulate
29 30 --     commands via their command number rather than their menu and item
29 31 --     numbers.
29 32 --
29 33 --     Each command can be assigned an integer command number from 1 to
29 34 --     32767. To associate a command number with a menu item, menu resources
29 35 --     are defined with the 'cmnu' resource type--not the 'MENU' type. The
29 36 --     'cmnu' type is just like the 'MENU' type, with an additional command
29 37 --     number field for each menu item. The program PostRez converts the
29 38 --     'cmnu' output of Rez into equivalent 'MENU' resources and one 'mntb'
29 39 --     resource. The 'mntb' resource maps command numbers to menu/item pairs.
29 40 --
29 41 --     The items of some menus cannot be determined until run-time. The font
29 42 --     menu is an example. In such cases, no command number is assigned by
29 43 --     you. Instead, the command number is equal to -(256 * menu + item).
29 44 --
29 45 --     The procedure CmdToMenuItem converts a command number to a menu id
29 46 --     and item number by the following method:
29 47 --
29 48 --     - If the command number is positive, the command table is searched
29 49 --       for the number. If it is found, CmdToMenuItem returns the corre-
29 50 --       sponding menu id and item number. Otherwise the menu and item are
29 51 --       zero.
29 52 --
29 53 --     - If the command number is negative, the menu number is the upper eight
29 54 --       bits, and the item number the lower eight bits, of the absolute value
29 55 --       of the command number. (This implies that menu id's must be <= 127
29 56 --       and item numbers must be <= 255.)
29 57 --
29 58 --     The function CmdFromMenuItem converts menu/items to command numbers by
29 59 --     the following method:
29 60 --
29 61 --     - If the item number is positive, the command table is search for the
29 62 --       given menu/item pair. If found, the corresponding command number is
29 63 --       returned. If not found, the number returned is equal to
29 64 --       -(256 * menu + item).
29 65 --
29 66 --     - If the item number is negative, the number returned is equal to
29 67 --       -item number.
29 68 --
29 69 --     Note that CmdToMenuItem and CmdFromMenuItem work regardless of whether
29 70 --     the command is actually installed in the menu bar. MacApp takes
29 71 --     advantage of this feature by having a set of generic "buzz" commands,
29 72 --     whose primary use is to set the name of the Undo command (e.g. 'Undo
29 73 --     Drawing').
29 74 --
29 75 --     The second feature of this unit is to optimize menu setups (changing
29 76 --     the appearance of menus and items). Normally, each call to CheckItem,
29 77 --     SetItemStyle and SetCmdIcon in turn calls CalcMenuSize because the width
29 78 --     or height of the menu may have changed. If menu setups are done all at
29 79 --     one time, then CalcMenuSize can be deferred until the end of the setup
29 80 --     process. The procedure PerformMenuSetup implements this mechanism. It
29 81 --     relies on the fact that menu setups are done all at once, and that you
29 82 --     will call SetCmdName, SetItemStyle and SetCmdIcon instead of SetItem,
29 83 --     SetItemStyle and SetCmdIcon. It works only for menus whose id is from
29 84 --     1 to mLastMenu. Menus with IDs greater than mLastMenu are not affected
29 85 --     by this scheme.
29 86 --
29 87 --     PerformMenuSetup accepts one parameter, a procedure which actually
29 88 --     implements the menu appearance changes. SetupMenus performs the
29 89 --     following steps:
29 90 --
29 91 --     - Before calling your menu setup procedure, StartupMenuSetup is called.
29 92 --       It disables all menus and items, remembers the current
29 93 --       menu proc and enabled flags of each menu, and sets the MenuProc of
29 94 --       each menu to gHNullMenuProc, thereby disabling CalcMenuSize.
29 95 --     - Your procedure is called. It in turn calls SetCmdName,
29 96 --       SetCmdStyle, SetCmdIcon, Enable, EnableCheck, or a
29 97 --       Toolbox routine (provided it isn't SetItem, SetItemStyle or
29 98 --       SetCmdIcon).
29 99 --     - Menus with at least one enabled item are enabled, CalcMenuSize is

```

```

29 99 --      called for those menus that need it, each menu's menuproc is restored,
29 100 --      and DrawMenuBar is called if the enabled state of any menu changed.
29 101 --      }
29 102 --
29 103 --      INTERFACE
29 104 --
29 105 --      {-----+
29 106 --      | Uses
29 107 --      +-----+}
29 108 --      USES
29 109 --          ($LOAD MacIntf.LOAD)
29 110 --          MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
29 111 --          ($LOAD)
29 112 --          UMAUtil,
29 113 --          UFailure,
29 114 --          UMemory
29 115 --      ($IFC qTrace)
29 116 --          , UTrace
29 117 --      ($ENDC)
29 118 --      ;
29 119 --
29 120 --      {-----+
29 121 --      | Global constants
29 122 --      +-----+}
29 123 --      CONST
29 124 --          KIDMNTBbyCmdNumber = 0; { menu command number table }
29 125 --
29 126 --      { Predefined menu IDs;
29 127 --      MacApp reads in menus with IDs [mApple..mLastInstalled], until a break
29 128 --      in the sequence, and installs them in the menu bar.
29 129 --      MacApp reads in menus with IDs[mLastInstalled+1..mLastMenu], until a
29 130 --      break in the sequence, but does not install them.
29 131 --
29 132 --      MacApp appends the desk accessories to the mApple menu. }
29 133 --
29 134 --      mApple      = 1; { MacApp does not disable any commands in
29 135 --                      this menu }
29 136 --      mFile       = 2;
29 137 --      mEdit       = 3;
29 138 --
29 139 --      mDebug      = 900; { read in if debugging is turned on; number
29 140 --                      corresponds with debugging command
29 141 --                      numbers }
29 142 --      mLastMenu   = 63; { MacApp disables menu titles
29 143 --                      mFile..mLastMenu generically }
29 144 --
29 145 --      {-----+
29 146 --      | Global types
29 147 --      +-----+}
29 148 --      TYPE
29 149 --
29 150 --          CmdNumber = INTEGER;
29 151 --
29 152 --      {-----+
29 153 --      | Global variables
29 154 --      +-----+}
29 155 --      VAR
29 156 --          gMenusAreSetup: BOOLEAN; { set FALSE after every event; set
29 157 --                                  TRUE by SetupTheMenus, which is called at
29 158 --                                  Idle Begin; if your DoIdle changes the
29 159 --                                  target or makes other changes that would
29 160 --                                  alter the appearance of menus, you must set
29 161 --                                  gMenusAreSetup to FALSE there. }
29 162 --          gRedrawMenuBar: BOOLEAN; { if TRUE, then DrawMenuBar will be
29 163 --                                  called by TApplication.SetupTheMenus. If
29 164 --                                  you have menus that are not handled by
29 165 --                                  MacApp, your implementation(s) of
29 166 --                                  DoSetupMenus can set this to TRUE to force
29 167 --                                  the menu bar to be redrawn. }
29 168 --          gHNullMenuProc: Handle; { Handle to null menu proc }
29 169 --      ($IFC qTrace)
29 170 --          gTraceSetupMenus: BOOLEAN;
29 171 --      ($ENDC)
29 172 --
29 173 --      {-----+
29 174 --      | Global routines
29 175 --      +-----+}
29 176 --      { The following routines all recognize command numbers of the form
29 177 --      -(256 * menu + item) and -(256 * menu) }
29 178 --      { The former is used when there is no command number; the latter to
29 179 --      enable/disable a whole menu (rare) }
29 180 --
29 181 --      FUNCTION CmdFromMenuItem(menu, item: INTEGER): CmdNumber;
29 182 --      { Returns the command number for the given menu ID and item number.
29 183 --
29 184 --      If the item number is positive, the command table is search for the
29 185 --      given menu/item pair. If found, the corresponding command number
29 186 --      is returned. If not found, the number returned is equal to
29 187 --      -(256 * menu + item).
29 188 --
29 189 --      If the item number is negative, the number returned is equal to
29 190 --      -item number.
29 191 --      }
29 192 --
29 193 --      PROCEDURE CmdToMenuItem(aCmd: CmdNumber; VAR menu, item: INTEGER);
29 194 --      { CmdToMenuItem returns the menu ID and item number of the given
29 195 --      command.
29 196 --
29 197 --      If aCmd is positive, the command table is searched for aCmd.

```

```

29 198 --      If it is found, CmdToMenuItem returns the corresponding menu ID and
29 199 --      item number. Otherwise menu and item are set to zero.
29 200 --
29 201 --      If aCmd is negative, the menu number is the upper eight bits, and
29 202 --      the item number the lower eight bits, of the absolute value of aCmd.
29 203 --      (This implies that menu id's must be <= 127 and item numbers must be
29 204 --      <= 255.)
29 205 --    }
29 206 --
29 207 --    PROCEDURE CmdToName(aCmd: CmdNumber; VAR menuText: Str255);
29 208 --    { Return the text of the menu item for the given command. }
29 209 --
29 210 --    PROCEDURE EachMenuDo (PROCEDURE DoToMenu(aMenuHandle: MenuHandle);
29 211 --      includeHierarchical: BOOLEAN);
29 212 --    { Sequences thru each menu in menu bar with ID in [0..mLastMenu], calling
29 213 --      DoToMenu for each menu. Handles mDebug as a special case. }
29 214 --
29 215 --    PROCEDURE Enable(aCmd: CmdNumber; canDo: BOOLEAN);
29 216 --    { Enable or disable a command }
29 217 --
29 218 --    PROCEDURE EnableCheck(aCmd: CmdNumber; canDo: BOOLEAN; checkIt: BOOLEAN);
29 219 --    { Enable/Disable and check/uncheck a command }
29 220 --
29 221 --    FUNCTION GetResMenu(menuResID: INTEGER): MenuHandle;
29 222 --    { Utility that just does GetResource('MENU', resourceID), so that it
29 223 --      works for menus not currently in the menu bar. (You cannot call
29 224 --      GetMenu more than once for the same menu, so you can use this
29 225 --      instead.) }
29 226 --
29 227 --    PROCEDURE InitUMenuSetup;
29 228 --    { Initializes this unit, and must be called before using any other part of
29 229 --      this unit. Loads the command table and sets up the null menu proc. }
29 230 --
29 231 --    PROCEDURE MAInsertMenu (theMenu: MenuHandle; beforeID: INTEGER);
29 232 --    { Equivalent to the Menu Manager's InsertMenu with the addition that the
29 233 --      MAInsertMenu looks for an 'mctb' resource whose id is the menu's id, and
29 234 --      if it is present applies the 'mctb' to the menu.}
29 235 --
29 236 --    PROCEDURE NeedCalcMenuSize(aMenuHandle: MenuHandle);
29 237 --    { This procedure is called as part of the menu setup process and
29 238 --      indicates that the given menu needs CalcMenuSize. You must
29 239 --      explicitly call this if you use change a menu by means other than
29 240 --      those supplied in this unit (e.g. you insert or delete items from
29 241 --      a menu). Failure to do so will result in improperly sized menus. }
29 242 --
29 243 --    PROCEDURE SetCmdIcon(aCmd: CmdNumber; menuIcon: Byte);
29 244 --    { You should call this (instead of the Toolbox's SetCmdIcon) for menus
29 245 --      that are in the range [1..mLastMenu]. In addition to changing the
29 246 --      menu this tells MacApp that the menu's size needs to be recomputed. }
29 247 --
29 248 --    PROCEDURE SetCmdName(aCmd: CmdNumber; menuText: Str255);
29 249 --    { You should call this (instead of the Toolbox's SetItem) for menus that
29 250 --      are in the range [1..mLastMenu]. In addition to changing the menu
29 251 --      this tells MacApp that the menu's size needs to be recomputed. }
29 252 --
29 253 --    PROCEDURE SetIndCmdName(aCmd: CmdNumber; rsrcID, strIndex: INTEGER);
29 254 --    { Does a GetIndString with the rsrcID & strIndex; and then a
29 255 --      SetCmdName. }
29 256 --
29 257 --    PROCEDURE SetMenuState(aCmd: CmdNumber;
29 258 --      rsrcID, falseBuzzItem, trueBuzzItem: INTEGER; stateVariable: BOOLEAN);
29 259 --    { IF stateVariable THEN
29 260 --      SetIndCmdName(aCmd, rsrcID, trueBuzzItem)
29 261 --    ELSE
29 262 --      SetIndCmdName(aCmd, rsrcID, falseBuzzItem);
29 263 --    }
29 264 --
29 265 --    PROCEDURE SetStyle(aCmd: CmdNumber; aStyle: Style);
29 266 --    { You should call this (instead of the Toolbox's SetItemStyle) for menus
29 267 --      that are in the range [1..mLastMenu]. In addition to changing the
29 268 --      menu this tells MacApp that the menu's size needs to be recomputed. }
29 269 --
29 270 --    PROCEDURE PerformMenuSetup (PROCEDURE TheMenuSetterUpper);
29 271 --
29 272 --    IMPLEMENTATION
29 273 --
29 274 --    {$I UMenuSetup.incl.p}

```

```

30 1 -- { $P }
30 2 -- { UMenuSetup.incl.p }
30 3 -- { Copyright © 1984-1988 by Apple Computer Inc. All rights reserved. }
30 4 --
30 5 -- {-----+
30 6 -- | Local types
30 7 -- +-----}
30 8 -- TYPE
30 9 -- { the following must match the declaration in the PostRez tool }
30 10 -- MenuCmdRecord =
30 11 -- RECORD
30 12 --     theCmdNumber:    INTEGER;
30 13 --     theMenuNumber:   INTEGER;
30 14 --     theItemNumber:   INTEGER;
30 15 -- END;
30 16 --
30 17 -- CmdArray = ARRAY[1..4000] OF MenuCmdRecord;
30 18 -- PCmdArray = ^CmdArray;
30 19 -- CmdTable = ^PCmdArray;
30 20 --
30 21 -- {-----+
30 22 -- | Local variables
30 23 -- +-----}
30 24 -- VAR
30 25 --     gCmdTable:    CmdTable;    { maps CmdNumber <--> (menu,item) }
30 26 --     gSizeCmdTable:  INTEGER;    { number of records in gCmdTable }
30 27 --
30 28 --
30 29 -- {-----+
30 30 -- | Forward declarations
30 31 -- +-----}
30 32 -- A PROCEDURE NullMenuProc(message: INTEGER; aMenuHandle: MenuHandle;
30 33 --     VAR menuRect: Rect; hitPt: Point;
30 34 --     VAR whichItem: INTEGER); FORWARD;
30 35 --
30 36 --
30 37 -- (*****
30 38 -- (* Global Routines
30 39 -- (*****
30 40 -- {-----+
30 41 -- | GetMenuList
30 42 -- +-----}
30 43 -- A FUNCTION GetMenuList: Handle; INLINE $2EB8, $0A1C;
30 44 -- {
30 45 --     MOVE.L    $A1C, (SP)
30 46 -- }
30 47 --
30 48 --
30 49 -- { $MAREs } { must be in resident segment because we mark what ever segment
30 50 --     contains this function resident }
30 51 -- { $IFC qTrace } { $D+ } { $ENDC }
30 52 -- {-----+
30 53 -- | CmdFromMenuItem
30 54 -- +-----}
30 55 -- A FUNCTION CmdFromMenuItem(menu, item: INTEGER): CmdNumber;
30 56 -- { Given a menuID/item # return the appropriate command number. IF there
30 57 -- is no such command number, return -BOR(BSL(menu, 8), item). If the
30 58 -- item number is <0 then assume that it is a negative command number. }
30 59 --
30 60 -- VAR
30 61 --     i:    INTEGER;
30 62 --     p:    PCmdArray;
30 63 -- A BEGIN
30 64 --     IF item < 0 THEN
30 65 --         CmdFromMenuItem := -item
30 66 --     ELSE
30 67 --         BEGIN
30 68 --             IF item > 0 THEN
30 69 --                 BEGIN
30 70 --                     p := gCmdTable^;
30 71 --
30 72 --                     FOR i := 1 TO gSizeCmdTable DO
30 73 --                         WITH p^[i] DO
30 74 --                             IF (menu = theMenuNumber) AND (item = theItemNumber) THEN
30 75 --                                 BEGIN
30 76 --                                     CmdFromMenuItem := theCmdNumber;
30 77 --                                     Exit(CmdFromMenuItem);
30 78 --                                 END;
30 79 --                             END;
30 80 --                         END;
30 81 --                     IF (menu > 127) OR (item > 255) THEN
30 82 --                         BEGIN
30 83 --                             Writeln('menu = ', menu:1, ' item = ', item:1);
30 84 --                             Writeln('Menu/item number is too big for a negative command number!');
30 85 --                             ProgramBreak('Try using a negative item number instead.');
```

```

30 99 -- +-----}
30 100 -- A  PROCEDURE CmdToMenuItem(aCmd: CmdNumber; VAR menu, item: INTEGER);
30 101 -- VAR
30 102 --     i:      INTEGER;
30 103 --     p:      PCmdArray;
30 104 --     low:     INTEGER;
30 105 --     high:    INTEGER;
30 106 --     test:    INTEGER;
30 107 0- A  BEGIN
30 108 --     IF aCmd < 0 THEN
30 109 1- BEGIN
30 110 --         menu := BSR(-aCmd, 8);
30 111 --         item := BAND(-aCmd, 255);
30 112 1- END
30 113 --     ELSE
30 114 1- BEGIN
30 115 --         low := 1;
30 116 --         high := gSizeCmdTable;
30 117 --         { Loop Invariant: unsearched range is [low..high] inclusive }
30 118 --
30 119 --         p := gCmdTable^;
30 120 --         WHILE low <= high DO
30 121 2- BEGIN
30 122 --             test := BSR(low + high, 1); { (low + high) DIV 2 }
30 123 --             WITH p^[test] DO
30 124 --                 IF aCmd = theCmdNumber THEN
30 125 3- BEGIN
30 126 --                     menu := theMenuNumber;
30 127 --                     item := theItemNumber;
30 128 --                     Exit(CmdToMenuItem);
30 129 3- END
30 130 --                 ELSE IF aCmd < theCmdNumber THEN
30 131 --                     high := test - 1
30 132 --                 ELSE
30 133 --                     low := test + 1;
30 134 2- END;
30 135 --
30 136 --         { not found }
30 137 --         menu := 0;
30 138 --         item := 0;
30 139 1- END;
30 140 0- A  END;
30 141 --     {$IFC qTrace}{$D++}{$ENDC}
30 142 --
30 143 --
30 144 --     {$S MRes}
30 145 --     {-----+
30 146 --     | CmdToName |
30 147 --     +-----}
30 148 -- A  PROCEDURE CmdToName(aCmd: CmdNumber; VAR menuText: Str255);
30 149 -- VAR
30 150 --     aMenu:      INTEGER;
30 151 --     anItem:     INTEGER;
30 152 --     mHandle:    MenuHandle;
30 153 0- A  BEGIN
30 154 --     menuText := '';
30 155 --
30 156 --     CmdToMenuItem(aCmd, aMenu, anItem);
30 157 --     mHandle := GetResMenu(aMenu);
30 158 --     IF mHandle <> NIL THEN
30 159 --         GetItem(mHandle, anItem, menuText);
30 160 0- A  END;
30 161 --
30 162 --
30 163 --     {-----+
30 164 --     | EachMenuDo |
30 165 --     +-----}
30 166 -- A  PROCEDURE EachMenuDo (PROCEDURE DoToMenu(aMenuHandle: MenuHandle);
30 167 --     includeHierarchical: BOOLEAN);
30 168 --     { Calls DoToMenu for each menu in the menu bar with ID, including the
30 169 --       hierarchical and popup menus if includeHierarchical is TRUE. }
30 170 --
30 171 -- TYPE
30 172 --     { See Inside Mac Volume 5, Menu Manager, 'Menu Mgr Data Structures' }
30 173 --
30 174 --     MLRecord = RECORD
30 175 --         mHandle:    MenuHandle;
30 176 --         rightCD:    INTEGER;
30 177 --     END;
30 178 --
30 179 --     HMenuList = ^PMenuList;
30 180 --     PMenuList = ^AMenuList;
30 181 --     AMenuList = RECORD
30 182 --         sizeX6:      INTEGER; { # of menus * 6 }
30 183 --         rightCd:     INTEGER;
30 184 --         notUsed:     INTEGER;
30 185 --         { Variable number of menus }
30 186 --         menus:       ARRAY[1..1000] OF MLRecord;
30 187 --         { Variable number of hierarchical menus follows }
30 188 --     END;
30 189 --
30 190 -- VAR
30 191 --     i:      INTEGER;
30 192 --     ml:      HMenuList;
30 193 --     noOfMenus: INTEGER;
30 194 --     menu:     MenuHandle;
30 195 --     noOfHierMenus: INTEGER;
30 196 --     hMenuPtr: LongInt;
30 197 --     lastHierMenu: INTEGER;
30 198 --     menurec: MLRecord;

```



```

30 198 0- A BEGIN
30 199 -- ml := HMenuList(GetMenuList);
30 200 -- noOfMenus := ml^.sizeX6 DIV 6;
30 201 --
30 202 -- FOR i := 1 TO noOfMenus DO
30 203 1- BEGIN
30 204 -- menu := ml^.menus[i].mHandle;
30 205 -- {$IFC qDebug}
30 206 -- IF (menu^.menuID = mDebug) THEN
30 207 2- BEGIN
30 208 -- menu^.menuID := 0; { store info about debug menu in index 0 }
30 209 -- DoToMenu(menu);
30 210 -- menu^.menuID := mDebug;
30 211 -2 END
30 212 -- ELSE
30 213 -- {$ENDC}
30 214 -- DoToMenu(menu);
30 215 -1 END;
30 216 --
30 217 -- { Now do hierarchial menus }
30 218 -- IF gConfiguration.hasHierarchicalMenus THEN
30 219 1- BEGIN
30 220 -- { Get pointer to first hierarchical menu }
30 221 -- HMenuPtr := ORD4(ml^) + ml^.sizeX6 + 12;
30 222 --
30 223 -- { get offset from HMenuPtr to last hierarchical menu }
30 224 -- lastHierMenu := PInteger(ORD4(ml^) + ml^.sizeX6 + 6)^;
30 225 -- noOfHierMenus := lastHierMenu DIV 6;
30 226 -- FOR i := 1 TO noOfHierMenus DO
30 227 2- BEGIN
30 228 -- BlockMove(Pointer(HMenuPtr), @menuRec, 6);
30 229 -- WITH menuRec DO
30 230 -- DoToMenu(mHandle);
30 231 -- HMenuPtr := HMenuPtr + 6;
30 232 -2 END;
30 233 -1 END;
30 234 --
30 235 -0 A END;
30 236 --
30 237 --
30 238 -- {$S MAREs}
30 239 -- {$IFC qTrace}{$D+}{$ENDC}
30 240 -- {-----+
30 241 -- | Enable |
30 242 -- +-----}
30 243 -- A PROCEDURE Enable(aCmd: CmdNumber; canDo: BOOLEAN);
30 244 -- VAR
30 245 -- menu, item: INTEGER;
30 246 -- aMenuHandle: MenuHandle;
30 247 0- A BEGIN
30 248 -- {$IFC qTrace}
30 249 -- IF gTracing AND gTraceSetupMenus THEN
30 250 -- WriteLn('..... Enable(', aCmd:1, ', ', gBoolString(canDo, '));
30 251 -- {$ENDC}
30 252 -- CmdToMenuItem(aCmd, menu, item);
30 253 -- aMenuHandle := GetMHandle(menu);
30 254 -- IF aMenuHandle <> NIL THEN
30 255 -- IF canDo THEN
30 256 -- EnableItem(aMenuHandle, item)
30 257 -- ELSE
30 258 -- DisableItem(aMenuHandle, item);
30 259 -0 A END;
30 260 -- {$IFC qTrace}{$D++}{$ENDC}
30 261 --
30 262 --
30 263 -- {$S MAREs}
30 264 -- {$IFC qTrace}{$D+}{$ENDC}
30 265 -- {-----+
30 266 -- | EnableCheck |
30 267 -- +-----}
30 268 -- A PROCEDURE EnableCheck(aCmd: CmdNumber; canDo, checkIt: BOOLEAN);
30 269 -- VAR
30 270 -- menu, item: INTEGER;
30 271 -- aMenuHandle: MenuHandle;
30 272 0- A BEGIN
30 273 -- {$IFC qTrace}
30 274 -- IF gTracing AND gTraceSetupMenus THEN
30 275 -- WriteLn('..... EnableCheck(', aCmd:1, ', ', gBoolString(canDo, ' ', gBoolString(checkIt, '));
30 276 -- {$ENDC}
30 277 -- CmdToMenuItem(aCmd, menu, item);
30 278 -- aMenuHandle := GetMHandle(menu);
30 279 -- IF aMenuHandle <> NIL THEN
30 280 1- BEGIN
30 281 -- IF canDo THEN
30 282 -- EnableItem(aMenuHandle, item)
30 283 -- ELSE
30 284 -- DisableItem(aMenuHandle, item);
30 285 -- CheckItem(aMenuHandle, item, checkIt);
30 286 -1 END;
30 287 -0 A END;
30 288 -- {$IFC qTrace}{$D++}{$ENDC}
30 289 --
30 290 --
30 291 -- {$S MAREs}
30 292 -- {-----+
30 293 -- | GetResMenu |
30 294 -- +-----}
30 295 -- A FUNCTION GetResMenu(menuResID: INTEGER): MenuHandle;
30 296 0- A BEGIN

```

```

30 297 --      GetResMenu := MenuHandle(GetResource('MENU', menuResID));
30 298 -0 A  END;
30 299 --
30 300 --
30 301 --      {$$ MAINit}
30 302 --      {-----+
30 303 --      |  InitMenuSetup
30 304 --      +-----}
30 305 -- A  PROCEDURE InitMenuSetup;
30 306 --
30 307 --      TYPE
30 308 --          PJMP    =      ^JmpRec;
30 309 --          JmpRec  =      RECORD
30 310 --                          opcode: INTEGER;
30 311 --                          target: Ptr;
30 312 --          END;
30 313 --
30 314 --      {$IFC qDebug}
30 315 --      VAR
30 316 --          i:                INTEGER;
30 317 --          previousCmdNumber: INTEGER;
30 318 --          aString:          Str255;
30 319 --      {$ENDC}
30 320 --
30 321 0- A  BEGIN
30 322 --          { read in the command table }
30 323 --          gCmdTable := CmdTable(GetResource('mntb', kIDMNTBbyCmdNumber));
30 324 --          FailNIL(gCmdTable);
30 325 --          gSizeCmdTable := GetHandleSize(Handle(gCmdTable)) DIV SIZEOF(MenuCmdRecord);
30 326 --
30 327 --      {$IFC qDebug}
30 328 --          previousCmdNumber := MAXINT;
30 329 --          FOR i := 1 TO gSizeCmdTable DO
30 330 1-      BEGIN
30 331 --              IF gCmdTable^[i].theCmdNumber = previousCmdNumber THEN
30 332 2-      BEGIN
30 333 --                  NumToString(previousCmdNumber, aString);
30 334 --                  ProgramBreak(CONCAT('Command #', aString, ' is duplicated in the command table.'));
30 335 -2      END;
30 336 --                  previousCmdNumber := gCmdTable^[i].theCmdNumber;
30 337 -1      END;
30 338 --      {$ENDC}
30 339 --
30 340 --          gHNullMenuProc := NewPermHandle(6);
30 341 --          FailNIL(gHNullMenuProc);
30 342 --          WITH PJMP(gHNullMenuProc)^ DO
30 343 1-      BEGIN
30 344 --              opcode := $4EF9;          { JMP }
30 345 --              target := @NullMenuProc;
30 346 -1      END;
30 347 --          { expect 68020 cache to be flushed before this is used }
30 348 -0 A  END { InitMenuSetup };
30 349 --
30 350 --
30 351 --      {$$ MRes}
30 352 --      {-----+
30 353 --      |  MAInsertMenu
30 354 --      +-----}
30 355 -- A  PROCEDURE MAInsertMenu (theMenu: MenuHandle; beforeID: INTEGER);
30 356 --      TYPE
30 357 --          MenuCRsrc = RECORD
30 358 --              numEntries:  INTEGER;
30 359 --              data:        ARRAY [1..1092] OF MEntry;
30 360 --          END;
30 361 --          PMenuCRsrc = ^MenuCRsrc;
30 362 --          HMenuCRsrc = ^PMenuCRsrc;
30 363 --      VAR
30 364 --          theColorRsrc:    HMenuCRsrc;
30 365 0- A  BEGIN
30 366 --          InsertMenu(theMenu, beforeID);
30 367 --          { Since only GetMenu automatically loads the appropriate color information,
30 368 --            and (sigh) since we can only call GetMenu once, and if you call DeleteMenu
30 369 --            all the good color stuff goes away (double sigh) we'll have to help out
30 370 --            the Menu Manager by doing its job for it }
30 371 --          IF gConfiguration.hasColorQD THEN
30 372 1-      BEGIN
30 373 --              theColorRsrc := HMenuCRsrc(GetResource('mctb', theMenu^.menuID));
30 374 --              IF theColorRsrc <> NIL THEN
30 375 2-      BEGIN
30 376 --                  HLock(Handle(theColorRsrc));
30 377 --                  WITH theColorRsrc^ DO
30 378 --                      SetMCEnties(numEntries, @data[1]);
30 379 --                  HUnlock(Handle(theColorRsrc));
30 380 --                  ReleaseResource(Handle(theColorRsrc));
30 381 -2      END;
30 382 -1      END;
30 383 -0 A  END;
30 384 --
30 385 --
30 386 --      {$$ MRes}
30 387 --      {-----+
30 388 --      |  NeedCalcMenuSize
30 389 --      +-----}
30 390 -- A  PROCEDURE NeedCalcMenuSize(aMenuHandle: MenuHandle);
30 391 0- A  BEGIN
30 392 --          WITH aMenuHandle^ DO
30 393 --              IF menuProc = gHNullMenuProc THEN
30 394 --                  menuWidth := 0;
30 395 -0 A  END;

```

```

30 396 --
30 397 --
30 398 --      ($$ Main)
30 399 --      { a null menuProc that is used to inhibit re-calculating the menu's size after each
30 400 --        call to EnableItem, CheckItem, etc. }
30 401 --      ($IFC qTrace){$D+}{$ENDC}
30 402 --      {-----+
30 403 --      |   NullMenuProc
30 404 --      +-----}
30 405 -- A  PROCEDURE NullMenuProc(message: INTEGER; aMenuHandle: MenuHandle; VAR menuRect: Rect;
30 406 --        hitPt: Point; VAR whichItem: INTEGER);
30 407 0- A  BEGIN
30 408 --        aMenuHandle^.menuWidth := 0;
30 409 -0 A  END;
30 410 --      ($IFC qTrace){$D+}{$ENDC}
30 411 --
30 412 --
30 413 --      ($$ MRes)
30 414 --      {-----+
30 415 --      |   PerformMenuSetup
30 416 --      +-----}
30 417 -- A  PROCEDURE PerformMenuSetup (PROCEDURE TheMenuSetterUpper);
30 418 --
30 419 --      TYPE
30 420 --          EnableArray = ARRAY [0..mLastMenu] OF BOOLEAN;      { Saved enable flags }
30 421 --          SavePrArray = ARRAY [0..mLastMenu] OF Handle;      { Saved menu procs }
30 422 --
30 423 --      VAR
30 424 --          aMenuHandle:   MenuHandle;
30 425 --          wasEnabled:     EnableArray;
30 426 --          savedProcs:     SavePrArray;
30 427 --
30 428 --
30 429 --      {-----+
30 430 --      |   StartMenuSetup
30 431 --      +-----}
30 432 -- B  PROCEDURE StartMenuSetup(aMenuHandle: MenuHandle);
30 433 --      VAR
30 434 --          item:           INTEGER;
30 435 --          theCmd:         CHAR;
30 436 0- B  BEGIN
30 437 --          WITH aMenuHandle^^ DO
30 438 --              IF (menuID >= 0) & (menuID <= mLastMenu) & (menuID <> mApple) THEN
30 439 1-          BEGIN
30 440 --              { Remember the menu itself was enabled, and disable the menu
30 441 --                and all of its items. }
30 442 --              wasEnabled[menuID] := Odd(enableFlags);
30 443 --              enableFlags := 0;
30 444 --
30 445 --              { Save the menu's menuproc and set it to the NullMenuProc, so that
30 446 --                CalcMenuSize is disabled (will do CalcMenuSize at end of setup). }
30 447 --              savedProcs[menuID] := menuProc; { See comment below }
30 448 --              menuProc := gHNullMenuProc;
30 449 --
30 450 --              { Uncheck all items. }
30 451 --              FOR item := 1 TO CountMItems(aMenuHandle) DO
30 452 --                  { Make sure we don't check items with sub-menus }
30 453 --                  IF gConfiguration.hasHierarchicalMenus THEN
30 454 2-                  BEGIN
30 455 --                      GetItemCmd(aMenuHandle, item, theCmd);
30 456 --                      IF theCmd <> CHR($1B) THEN
30 457 --                          CheckItem(aMenuHandle, item, FALSE);
30 458 -2                  END
30 459 --                  ELSE
30 460 --                      CheckItem(aMenuHandle, item, FALSE);
30 461 -1                  END;
30 462 -0 B  END { StartMenuSetup };
30 463 --
30 464 --      {-----+
30 465 --      |   EndMenuSetup
30 466 --      +-----}
30 467 -- B  PROCEDURE EndMenuSetup (aMenuHandle: MenuHandle);
30 468 --      VAR
30 469 --          newFlags: LONGINT;
30 470 --          menuID:   INTEGER;
30 471 0- B  BEGIN
30 472 --          WITH aMenuHandle^^ DO
30 473 --              IF (menuID >= 0) & (menuID <= mLastMenu) & (menuID <> mApple) THEN
30 474 1-          BEGIN
30 475 --              newFlags := enableFlags;
30 476 --              { If any items are enabled, enable the menu }
30 477 --              IF newFlags <> 0 THEN
30 478 2-                  BEGIN
30 479 --                      newFlags := BOR(1, newFlags);
30 480 --                      enableFlags := newFlags;
30 481 -2                  END;
30 482 --
30 483 --              { If the menu's enabled state changed, we have to draw the menu bar. }
30 484 --              IF Odd(newFlags) <> wasEnabled[menuID] THEN
30 485 --                  gRedrawMenuBar := TRUE;
30 486 --
30 487 --              { Restore the menuproc. }
30 488 --              menuProc := savedProcs[menuID];
30 489 --
30 490 --              { menuWidth set to 0 by routines that require CalcMenuSize, by
30 491 --                calling NeedCalcMenu. }
30 492 --              IF menuWidth = 0 THEN
30 493 --                  CalcMenuSize(aMenuHandle);
30 494 -1          END;

```

```

30 495 -0 B      END { EndMenuSetup };
30 496 --
30 497 0- A      BEGIN
30 498 --          EachMenuDo(StartMenuSetup, TRUE);
30 499 --          TheMenuSetterUpper;
30 500 --          EachMenuDo(EndMenuSetup, TRUE);
30 501 --
30 502 --          IF gRedrawMenuBar THEN
30 503 1-              BEGIN
30 504 --                  DrawMenuBar;
30 505 --                  gRedrawMenuBar := FALSE;
30 506 -1              END;
30 507 --
30 508 --          gMenusAreSetup := TRUE;
30 509 -0 A      END { PerformMenuSetup };
30 510 --
30 511 --
30 512 --          {$$ MAREs}
30 513 --          {$IFC qTrace}{$D+}{$ENDC}
30 514 --          {-----+
30 515 --          | SetCmdIcon
30 516 --          +-----}
30 517 -- A      PROCEDURE SetCmdIcon(aCmd: CmdNumber; menuIcon: Byte);
30 518 --      VAR
30 519 --          menu, item:    INTEGER;
30 520 --          aMenuHandle:  MenuHandle;
30 521 0- A      BEGIN
30 522 --          {$IFC qTrace}
30 523 --          IF gTracing AND gTraceSetupMenus THEN
30 524 --              WriteLn('.... SetCmdIcon(' , aCmd:1, ', ', menuIcon:1, ')');
30 525 --          {$ENDC}
30 526 --          CmdToMenuItem(aCmd, menu, item);
30 527 --          aMenuHandle := GetResMenu(menu);
30 528 --          IF aMenuHandle <> NIL THEN
30 529 --              SetItemIcon(aMenuHandle, item, menuIcon);
30 530 -0 A      END;
30 531 --          {$IFC qTrace}{$D++}{$ENDC}
30 532 --
30 533 --
30 534 --          {$$ MAREs}
30 535 --          {$IFC qTrace}{$D+}{$ENDC}
30 536 --          {-----+
30 537 --          | SetCmdName
30 538 --          +-----}
30 539 -- A      PROCEDURE SetCmdName(aCmd: CmdNumber; menuText: Str255);
30 540 --      VAR
30 541 --          menu, item:    INTEGER;
30 542 --          aMenuHandle:  MenuHandle;
30 543 0- A      BEGIN
30 544 --          {$IFC qTrace}
30 545 --          IF gTracing AND gTraceSetupMenus THEN
30 546 --              WriteLn('.... SetCmdName(' , aCmd:1, ', ', menuText, ')');
30 547 --          {$ENDC}
30 548 --          CmdToMenuItem(aCmd, menu, item);
30 549 --          aMenuHandle := GetResMenu(menu);
30 550 --          IF aMenuHandle <> NIL THEN
30 551 --              SetItem(aMenuHandle, item, menuText);
30 552 -0 A      END;
30 553 --          {$IFC qTrace}{$D++}{$ENDC}
30 554 --
30 555 --
30 556 --          {$$ MAREs}
30 557 --          {$IFC qTrace}{$D+}{$ENDC}
30 558 --          {-----+
30 559 --          | SetIndCmdName
30 560 --          +-----}
30 561 -- A      PROCEDURE SetIndCmdName(aCmd: CmdNumber; rsrcID, strIndex: INTEGER);
30 562 --      VAR
30 563 --          s: Str255;
30 564 0- A      BEGIN
30 565 --          GetIndString(s, rsrcID, strIndex);
30 566 --          SetCmdName(aCmd, s);
30 567 -0 A      END;
30 568 --          {$IFC qTrace}{$D++}{$ENDC}
30 569 --
30 570 --
30 571 --          {$$ MAREs}
30 572 --          {$IFC qTrace}{$D+}{$ENDC}
30 573 --          {-----+
30 574 --          | SetMenuState
30 575 --          +-----}
30 576 -- A      PROCEDURE SetMenuState(aCmd: CmdNumber;
30 577 --          rsrcID, falseBuzzItem, trueBuzzItem: INTEGER; stateVariable: BOOLEAN);
30 578 --      VAR
30 579 --          buzzItem:    INTEGER;
30 580 0- A      BEGIN
30 581 --          IF stateVariable THEN
30 582 --              buzzItem := trueBuzzItem
30 583 --          ELSE
30 584 --              buzzItem := falseBuzzItem;
30 585 --
30 586 --          SetIndCmdName(aCmd, rsrcID, buzzItem);
30 587 -0 A      END;
30 588 --          {$IFC qTrace}{$D++}{$ENDC}
30 589 --
30 590 --
30 591 --          {$$ MAREs}
30 592 --          {$IFC qTrace}{$D+}{$ENDC}
30 593 --          {-----+

```

```
30 594 --      | SetStyle                      |
30 595 --      +-----+
30 596 -- A  PROCEDURE SetStyle(aCmd: CmdNumber; aStyle: Style);
30 597 --      VAR
30 598 --          menu, item:      INTEGER;
30 599 --          aMenuHandle:     MenuHandle;
30 600 0- A  BEGIN
30 601 --      { $IFC qTrace }
30 602 --          IF qTracing AND qTraceSetupMenus THEN
30 603 --              WriteLn('..... SetStyle(', aCmd:1, ', ', QByte(aStyle):1, ')');
30 604 --      { $ENDC }
30 605 --          CmdToMenuItem(aCmd, menu, item);
30 606 --          aMenuHandle := GetResMenu(menu);
30 607 --          IF aMenuHandle <> NIL THEN
30 608 --              SetItemStyle(aMenuHandle, item, aStyle);
30 609 -0 A  END;
30 610 --      { $IFC qTrace } { $D++ } { $ENDC }
29 275 --
29 276 --      END { UMenuSetup }.
```

[illegible]

```

31 99 --      {$ENDC}
31 100 --          END { TObject };
31 101 --
31 102 --
31 103 --      {-----+
31 104 --      | Global routines |
31 105 --      +-----}
31 106 --      PROCEDURE FreeObject(obj: TObject);
31 107 --      { IF obj <> NIL THEN obj.Free; useful for freeing an object that might
31 108 --        sometimes be NIL. }
31 109 --
31 110 --
31 111 --      {$IFC qDebug}
31 112 --      PROCEDURE IdUObject;
31 113 --      { Writeln UObject compile date }
31 114 --
31 115 --      PROCEDURE WritelnField (fieldName: Str255; fieldAddr: Ptr; fieldType: INTEGER);
31 116 --      { Writes the given field in the writeln window in the form
31 117 --        WRITE(fieldName, '-', valueAsString). }
31 118 --
31 119 --      PROCEDURE InspectObject(obj: Handle);
31 120 --      { Called by debugger to inspect an object. }
31 121 --
31 122 --      FUNCTION IsObject (obj: TObject): BOOLEAN;
31 123 --      { Returns true if obj references a real object, false if obj is NIL or
31 124 --        a non-object reference. }
31 125 --
31 126 --      PROCEDURE BuildObjNameTable;
31 127 --      { Builds the object name table. FindObjID can't be used until after
31 128 --        this has been called. }
31 129 --
31 130 --      FUNCTION FindObjID(anObjName: String8): INTEGER;
31 131 --      { Given an object name, return its id. Relies on gObjNameTable. }
31 132 --
31 133 --      PROCEDURE LookupObjName(objID: INTEGER; VAR objName: String8);
31 134 --      { Given an object id, return the object's name. Does not need
31 135 --        gObjNameTable. }
31 136 --      {$ENDC qDebug}
31 137 --
31 138 --
31 139 --      PROCEDURE OBJFail(error: INTEGER); { Called by UTrace }
31 140 --
31 141 --      {$%+      Enable '%' in identifiers}
31 142 --
31 143 --      PROCEDURE %Pgml;
31 144 --      { The Pascal compiler generates code to call this procedure before
31 145 --        the first unit is initialized. }
31 146 --
31 147 --      FUNCTION %OBCHK(ordObject, jumpTablePtr: LONGINT): LONGINT;
31 148 --      { Called before the first unit is ginitialized. This returns its first
31 149 --        argument IF it is NIL or passes a class-membership check; ELSE calls
31 150 --        OBJFail }
31 151 --
31 152 --      { Calls to following generated by compiler for New(obj) }
31 153 --      PROCEDURE %OBNEW(VAR obj: LONGINT; jumpTablePtr: LONGINT; itsSize: INTEGER);
31 154 --      { when type of obj in same unit }
31 155 --
31 156 --      PROCEDURE %_BDISP(VAR obj: Handle);
31 157 --
31 158 --      {$%-      Disable '%' in identifiers}
31 159 --
31 160 --      IMPLEMENTATION
31 161 --
31 162 --      {$%+      Enable '%' in identifiers}
31 163 --
31 164 --      {$I UObject.incl.p}

```

```

32 1 --      {$P}
32 2 --      { UObject.incl.p }
32 3 --      { Copyright © 1984-1988 by Apple Computer, Inc. All rights reserved. }
32 4 --
32 5 --
32 6 --      {-----+
32 7 --      | Local variables
32 8 --      +-----}
32 9 --      {$IFC qDebug}
32 10 --      VAR
32 11 --          gObjNameTable:   HObjNameTbl;   { handle to table of object names }
32 12 --          gInspectLinePos:  INTEGER;       { Used to do line breaks when inspecting fields. }
32 13 --      {$ENDC}
32 14 --
32 15 --
32 16 --      {$R-      Turn range checking off}
32 17 --
32 18 --      {$SETC qMacApp := TRUE}
32 19 --      {$I ObjLib.incl.p}
32 20 --      {ObjLib.incl.p}
32 21 --      {Copyright © 1984-1988 Apple Computer, Inc. All rights reserved.}
32 22 --
32 23 --      {This is an alternate library for Object Pascal, Assembler, etc users who
32 24 --      do not wish to use MacApp. MacApp users should use UObject instead}
32 25 --
32 26 --      {-----+
32 27 --      | External declarations
32 28 --      +-----}
32 29 --      A PROCEDURE OBJFail(error: INTEGER); EXTERNAL;
32 30 --      A FUNCTION %_InObj(ordObject, jumpTablePtr: LONGINT): BOOLEAN; EXTERNAL;
32 31 --      A PROCEDURE %_SetClassIndex(ordObject, jumpTablePtr: LONGINT); EXTERNAL;
32 32 --
32 33 --      A PROCEDURE %_RTS1; EXTERNAL;
32 34 --      A PROCEDURE %_InitObj; EXTERNAL;
32 35 --
32 36 --      {$SETC qInspector := TRUE}
32 37 --      {$IFC qInspector}
32 38 --      A PROCEDURE AddObjectToInspector (theObject: TObject); EXTERNAL;
32 39 --      A PROCEDURE RemoveObjectFromInspector (theObject: TObject); EXTERNAL;
32 40 --      {$ENDC}
32 41 --
32 42 --      (*-----+
32 43 --      | Global Routines
32 44 --      +-----*)
32 45 --      {$IFC qTrace}{$D+}{$ENDC}
32 46 --      {$S Main}
32 47 --      {-----+
32 48 --      | % Pgml
32 49 --      +-----}
32 50 --
32 51 --      {$J+}
32 52 --      VAR
32 53 --          %_METHCACHE:   Ptr;
32 54 --      {$J-}
32 55 --
32 56 --      A PROCEDURE % Pgml:
32 57 --          {The Pascal compiler generates code to call this procedure auto-
32 58 --          matically, before initializing the units and starting the
32 59 --          application's main program. This function must always work on 64K
32 60 --          ROMs.}
32 61 --      BEGIN
32 62 --          {Important note for understanding why this is here:
32 63 --
32 64 --          In an unoptimized program, %_InitObj gets called, which does nothing
32 65 --          but return. However, in an optimized program, the Linker changes
32 66 --          this to call %_InitOptObj, was does do something. So don't be
32 67 --          fooled into believing that the next statement is useless.}
32 68 --
32 69 --          %_MethCache := NewPtr (256 * 8); { Used for non-opt, thrown away in opt. }
32 70 --          %_InitObj; {A no-op for non-optimized code}
32 71 --      END (% Pgml);
32 72 --
32 73 --      {$IFC qMacApp}
32 74 --      {$S MAMain}
32 75 --      {$ENDC}
32 76 --      {-----+
32 77 --      | %_OBNEW
32 78 --      +-----}
32 79 --      A PROCEDURE %_OBNEW (VAR obj: LONGINT; jumpTablePtr: LONGINT; itsSize: INTEGER);
32 80 --          {$IFC qDebug}
32 81 --          CONST initVal = $3F21;
32 82 --
32 83 --          VAR i:   INTEGER; {must be first local and 2 bytes in size}
32 84 --              p:   PInteger;
32 85 --              n:   String8;
32 86 --              s:   Str255;
32 87 --          {$ENDC}
32 88 --
32 89 --      BEGIN
32 90 --          {$IFC qDebug}
32 91 --          IF gAskAboutAlloc AND CanReadLn THEN
32 92 --              BEGIN
32 93 --                  GetMethodName(%_GetA6+4, s);
32 94 --                  LookupObjName(jumpTablePtr-%_GetA5, n);
32 95 --                  Writeln('Within ', s, ', ', trying to make a '', n, '');

```



```

33 80 --          IF ReadYesNo('      Return NIL (Y or N) [N]? ') THEN
33 81 2-          BEGIN
33 82 --          obj := 0;
33 83 --          EXIT(%_OBNEW);
33 84 -2          END;
33 85 -1          END;
33 86 --          {$ENDC qDebug}
33 87 --
33 88 --          {$IFC qMacApp}
33 89 --          obj := ORD(NewPermHandle(itsSize));
33 90 --          {$ELSEC}
33 91 --          obj := ORD(NewHandle(itsSize));
33 92 --          {$ENDC}
33 93 --
33 94 --          IF obj <> 0 THEN
33 95 1-          BEGIN
33 96 --
33 97 --          {$IFC qDebug}
33 98 --          (Initialize the object to $3F213F21...)
33 99 --          p := PInteger(Handle(obj)^);
33 100 --          FOR i := 1 TO itsSize DIV 2 DO
33 101 2-          BEGIN
33 102 --          p^ := initVal;
33 103 --          p := PInteger(ORD(p) + 2);
33 104 -2          END;
33 105 --          {$ENDC}
33 106 --
33 107 --          (Install class index)
33 108 --          %_SetClassIndex(obj, jumpTablePtr);
33 109 --
33 110 --          {$IFC qInspector}
33 111 --          AddObjectToInspector(TObject(obj));
33 112 --          {$ENDC}
33 113 -1          END;
33 114 -0 A      END (%_OBNEW);
33 115 --
33 116 --
33 117 --          {$IFC qMacApp}
33 118 --          {$S MAMain}
33 119 --          {$ENDC}
33 120 --          {-----+
33 121 --          | %_OBDISP |
33 122 --          +-----}
33 123 -- A      PROCEDURE %_OBDISP(VAR obj: Handle);
33 124 0- A      BEGIN
33 125 --          DisposHandle(obj);
33 126 -0 A      END (%_OBDISP);
33 127 --
33 128 --
33 129 --          {$IFC qMacApp}
33 130 --          {$S MAMain}
33 131 --          {$ENDC}
33 132 --          {-----+
33 133 --          | %_OBCHK |
33 134 --          +-----}
33 135 -- A      FUNCTION %_OBCHK(ordObject, jumpTablePtr: LONGINT): LONGINT;
33 136 0- A      BEGIN
33 137 --          %_OBCHK := ordObject;
33 138 --          IF ordObject <> 0 THEN
33 139 --          IF NOT %_InObj(ordObject, jumpTablePtr) THEN
33 140 1-          BEGIN
33 141 --          OBJFail(kFailCoercion);
33 142 --          EXIT(%_OBCHK);
33 143 -1          END;
33 144 -0 A      END (%_OBCHK);
33 145 --          {$IFC qTrace}{SD+}{$ENDC}
33 146 --
33 147 --
33 148 --          (*****
33 149 --          (*      T O b j e c t
33 150 --          (*****
33 151 --          {$IFC qMacApp}
33 152 --          {$S MAMain}
33 153 --          {$ENDC}
33 154 --          {-----+
33 155 --          | ShallowClone |
33 156 --          +-----}
33 157 -- A      FUNCTION TObject.ShallowClone;
33 158 --
33 159 --          VAR result:      Handle;
33 160 --          oldPerm:      BOOLEAN;
33 161 --          err:          OSErr;
33 162 --
33 163 --          {$IFC qDebug}
33 164 --          n:          String8;
33 165 --          s:          Str255;
33 166 --          c:          String8;
33 167 --          m:          String8;
33 168 --          {$ENDC}
33 169 --
33 170 0- A      BEGIN
33 171 --          {$IFC qDebug}
33 172 --          IF gAskAboutAlloc THEN
33 173 1-          BEGIN
33 174 --          n := TypeName;
33 175 --
33 176 --          GetProcName(%_GetA6+4, c, m);
33 177 --
33 178 --          IF (c = 'TOBJECT ') AND (m = 'CLONE ') THEN (report about caller of TObject.Clone, instead)

```

```

33 179 --      GetProcName(PLongInt(%_GetA6)^ + 4, c, m);
33 180 --
33 181 --      IF c = kSpace8 THEN
33 182 --          s := m
33 183 --      ELSE
33 184 --          s := CONCAT(c, '.', m);
33 185 --
33 186 --      Writeln('Within ', s, ', trying to clone a '''', n, '''.');
33 187 --
33 188 --      IF ReadYesNo('      Return NIL (Y or N) [N]? ') THEN
33 189 2-      BEGIN
33 190 --          ShallowClone := NIL;
33 191 --          EXIT(ShallowClone);
33 192 -2      END;
33 193 -1      END;
33 194 --      {$ENDC}
33 195 --
33 196 --      result := Handle(SELF);
33 197 --
33 198 --      {$IFC qMacApp}
33 199 --          oldPerm := PermAllocation(TRUE);
33 200 --      {$ENDC}
33 201 --
33 202 --      err := HandToHand(result);
33 203 --
33 204 --      {$IFC qMacApp}
33 205 --          oldPerm := PermAllocation(oldPerm);
33 206 --      {$ENDC}
33 207 --
33 208 --      IF err <> noErr THEN
33 209 --          result := NIL
33 210 --      {$IFC qInspector}
33 211 --      ELSE
33 212 --          AddObjectToInspector(TObject(result))
33 213 --      {$ENDC}
33 214 --      ;
33 215 --
33 216 --      ShallowClone := TObject(result);
33 217 -0 A  END (TObject.ShallowClone);
33 218 --
33 219 --
33 220 --      {$IFC qMacApp}
33 221 --      {$S MAMain}
33 222 --      {$ENDC}
33 223 --      {-----+
33 224 --      | Clone |
33 225 --      +-----}
33 226 -- A  FUNCTION TObject.Clone;
33 227 0- A  BEGIN
33 228 --      Clone := ShallowClone;
33 229 -0 A  END (TObject.Clone);
33 230 --
33 231 --
33 232 --      {$IFC qMacApp}
33 233 --      {$S MAMain}
33 234 --      {$ENDC}
33 235 --      {-----+
33 236 --      | ShallowFree |
33 237 --      +-----}
33 238 -- A  PROCEDURE TObject.ShallowFree;
33 239 0- A  BEGIN
33 240 --      DisposHandle(Handle(SELF));
33 241 -0 A  END (TObject.ShallowFree);
33 242 --
33 243 --
33 244 --      {$IFC qMacApp}
33 245 --      {$S MAMain}
33 246 --      {$ENDC}
33 247 --      {-----+
33 248 --      | Free |
33 249 --      +-----}
33 250 -- A  PROCEDURE TObject.Free;
33 251 0- A  BEGIN
33 252 --      {$IFC qInspector}
33 253 --          RemoveObjectFromInspector(SELF);
33 254 --      {$ENDC}
33 255 --      ShallowFree;
33 256 -0 A  END (TObject.Free);
33 257 --
33 258 --
33 259 --      {$IFC qInspector}
33 260 --      {$S MAInspector}
33 261 --      {-----+
33 262 --      | GetInspectorName |
33 263 --      +-----}
33 264 -- A  PROCEDURE TObject.GetInspectorName (VAR inspectorName: Str255);
33 265 0- A  BEGIN
33 266 -0 A  END (TObject.GetInspectorName);
33 267 --      {$ENDC}
32 20 --
32 21 --
32 22 --      (*****
32 23 --      (* Global routines *)
32 24 --      (*****
32 25 --      {$IFC qMacApp AND qDebug}
32 26 --      {$IFC qTrace} {$D+} {$ENDC}
32 27 --      {$S MAInit}
32 28 --      {-----+
32 29 --      | BuildObjNameTable |

```

```

32 30 -- +-----}
32 31 -- A PROCEDURE BuildObjNameTable;
32 32 --
32 33 -- CONST
32 34 --     empty = '12345678';
32 35 --
32 36 -- TYPE
32 37 --     ZPPAOC8 = ^ZPAOC8;
32 38 --     ZPAOC8 = PACKED ARRAY[0..7] OF Char;
32 39 --
32 40 -- VAR
32 41 --     i:           INTEGER;
32 42 --     objRec:      ObjNameRec;
32 43 --     methTable:   Handle;
32 44 --     p:           PInteger;
32 45 --     backP:       PInteger;
32 46 --     endMeth:     PInteger;
32 47 --     offset:     LONGINT;
32 48 --
32 49 0- A BEGIN
32 50 --     gObjNameTable := HObjNameTbl(NewHandle(0));
32 51 --     methTable := GetNamedResource('CODE', '%_MethTables');
32 52 --     IF methTable <> NIL THEN
32 53 1- BEGIN
32 54 --         p := PInteger(methTable^);
32 55 --         endMeth := PInteger(Ord(p) + SizeResource(methTable));
32 56 --
32 57 --         WHILE Ord(p) <= Ord(endMeth)-8 DO
32 58 2- BEGIN
32 59 --             IF ZPPAOC8(p)^ = 'CLASINFO' THEN
32 60 3- BEGIN
32 61 --                 { we are at the end of the object info; search backwards }
32 62 --                 backP := p;
32 63 --                 objRec.objName := empty;
32 64 --
32 65 --                 { copy the object type name into the record }
32 66 --                 FOR i := 8 DOWNT0 1 DO
32 67 4- BEGIN
32 68 --                     backP := PInteger(Ord(backP) - 1);
32 69 --                     objRec.objName[i] := Chr(BAND(PByte(backP)^, 127));
32 70 4- END;
32 71 --
32 72 --                 { find the # of methods for this object type }
32 73 --                 backP := PInteger(Ord(backP) - 2);
32 74 --                 i := 0;
32 75 --                 WHILE i <> backP^ DO
32 76 4- BEGIN
32 77 --                     i := i + 1;
32 78 --                     backP := PInteger(Ord(backP) - 4);
32 79 4- END;
32 80 --
32 81 --                 backP := PInteger(Ord(backP) - 6);
32 82 --                 objRec.objID := backP^ + 2; { object ID is 2 more than jt offset }
32 83 --
32 84 --                 { add it to the list }
32 85 --                 offset := Munger(Handle(gObjNameTable), 0,
32 86 --                     NIL, 0, @objRec, SIZEOF(objRec));
32 87 --
32 88 --                 p := PInteger(Ord(p) + 8); { skip 'CLASINFO' }
32 89 3- END
32 90 --             ELSE
32 91 --                 p := PInteger(Ord(p) + 2);
32 92 2- END;
32 93 1- END;
32 94 0- A END { BuildObjNameTable };
32 95 --     { $IFC qTrace } { $D++ } { $ENDC }
32 96 --     { $ENDC qMacApp AND qDebug }
32 97 --
32 98 --
32 99 --     { $IFC qDebug }
32 100 --     { $IFC qTrace } { $D++ } { $ENDC }
32 101 --     { $S MAdDebug }
32 102 --     {-----+
32 103 --     | FindObjID |
32 104 --     +-----}
32 105 0- A FUNCTION FindObjID(anObjName: String8): INTEGER;
32 106 --     VAR i:           INTEGER;
32 107 0- A BEGIN
32 108 --         { pad objName to 8 chars and uppercase it }
32 109 --         anObjName := Copy(Concat(anObjName, kSpace8), 1, 8);
32 110 --         UpString8(anObjName);
32 111 --
32 112 --         i := GetHandleSize(Handle(gObjNameTable)) DIV SIZEOF(ObjNameRec);
32 113 --         WHILE i > 0 DO
32 114 1- BEGIN
32 115 --             WITH gObjNameTable^[i] DO
32 116 --                 IF anObjName = objName THEN
32 117 2- BEGIN
32 118 --                     FindObjId := objID;
32 119 --                     Exit(FindObjID);
32 120 2- END;
32 121 --                 i := i - 1;
32 122 1- END;
32 123 --
32 124 --         ProgramBreak(CONCAT('FindObjId: Can''t find object name ', anObjName));
32 125 --         FindObjID := 0;
32 126 --
32 127 0- A END { FindObjID };
32 128 --     { $IFC qTrace } { $D++ } { $ENDC }

```

```

32 129 --      {SENDC qDebug}
32 130 --
32 131 --
32 132 --      {$$ MAMain}
32 133 --      {$IFC qTrace}{$D+}{$SENDC}
32 134 --      {-----+
32 135 --      |   FreeObject
32 136 --      +-----}
32 137 -- A  PROCEDURE FreeObject(obj: TObject);
32 138 --
32 139 0- A  BEGIN
32 140 --      IF obj <> NIL THEN
32 141 --          obj.Free;
32 142 -0 A  END;
32 143 --      {$IFC qTrace}{$D+}{$SENDC}
32 144 --
32 145 --
32 146 --      {$IFC qDebug}
32 147 --      {$IFC qTrace}{$D+}{$SENDC}
32 148 --      {$$ MAInspect} { MacApp InitToolbox sets this segment resident }
32 149 --      {-----+
32 150 --      |   InspectField
32 151 --      +-----}
32 152 -- A  PROCEDURE InspectField (fieldName: Str255; fieldAddr: Ptr; fieldType: INTEGER);
32 153 --
32 154 --      VAR
32 155 --          s:      Str255;
32 156 --          x:      INTEGER;
32 157 --
32 158 0- A  BEGIN
32 159 --      {$IFC qWWNeeded} { Can't do this if there is no writeln window }
32 160 --      IF fieldType <> bClass THEN
32 161 1-      BEGIN
32 162 --          FieldToString(fieldAddr, fieldType, s);
32 163 --          x := LENGTH(fieldName) + 1 + LENGTH(s);
32 164 --          IF gInspectLinePos + x + 2 >= gWWPerLine THEN
32 165 2-      BEGIN
32 166 --          WRITELN;
32 167 --          gInspectLinePos := 0;
32 168 -2      END
32 169 --      ELSE IF gInspectLinePos <> 0 THEN { If not at the start of a line }
32 170 2-      BEGIN
32 171 --          WRITE(' ');
32 172 --          gInspectLinePos := gInspectLinePos + 2;
32 173 -2      END;
32 174 --          WRITE(fieldName, '=', s);
32 175 --          gInspectLinePos := gInspectLinePos + x;
32 176 -1      END;
32 177 --      {$SENDC}
32 178 -0 A  END;
32 179 --      {$IFC qTrace}{$D+}{$SENDC}
32 180 --      {$SENDC qDebug}
32 181 --
32 182 --
32 183 --      {$IFC qDebug}
32 184 --      {$IFC qTrace}{$D+}{$SENDC}
32 185 --      {$$ MAInspect} { MacApp InitToolbox sets this segment resident }
32 186 --      {-----+
32 187 --      |   InspectObject
32 188 --      +-----}
32 189 -- A  PROCEDURE InspectObject(obj: Handle);
32 190 --
32 191 --      VAR
32 192 --          hex:      String8;
32 193 --          x:      INTEGER;
32 194 0- A  BEGIN
32 195 --      IF IsObject(TObject(obj)) THEN
32 196 1-      BEGIN
32 197 --          x := 0;
32 198 --          HLock(Handle(obj));
32 199 --          TObject(obj).Inspect;
32 200 --          WRITELN;
32 201 --          HUnlock(Handle(obj));
32 202 -1      END
32 203 --      ELSE
32 204 1-      BEGIN
32 205 --          LintToHex(ORD(obj), hex, 6);
32 206 --          Writeln('$', hex, ' is not a TObject!');
32 207 -1      END;
32 208 -0 A  END { InspectObject };
32 209 --      {$IFC qTrace}{$D+}{$SENDC}
32 210 --      {$SENDC qDebug}
32 211 --
32 212 --
32 213 --      {$IFC qDebug}
32 214 --      {$IFC qTrace}{$D+}{$SENDC}
32 215 --      {$$ MAInspect}
32 216 --      {-----+
32 217 --      |   IsObject
32 218 --      +-----}
32 219 -- A  FUNCTION IsObject (obj: TObject): BOOLEAN;
32 220 --
32 221 --      VAR
32 222 0- A  BEGIN
32 223 --      IF NOT ODD(ORD4(obj)) THEN
32 224 1-      BEGIN
32 225 --          ptrToObject := @GetA5 + FindObjID('TObject');
32 226 --          IsObject := @_InObj(ORD4(obj), ptrToObject);
32 227 -1      END

```

```

32 228 --      ELSE
32 229 --          IsObject := FALSE;
32 230 -0 A      END { IsObject };
32 231 --      { $IFC qTrace } { $D++ } { $ENDC }
32 232 --      { $ENDC qDebug }
32 233 --
32 234 --
32 235 --      { $IFC qDebug }
32 236 --      { $IFC qTrace } { $D+ } { $ENDC }
32 237 --      { $S MAdEbug }
32 238 --      {-----+
32 239 --      |      LookupObjName          |      Given A5-offset return Object Type Name
32 240 --      +-----+
32 241 -- A      PROCEDURE LookupObjName (objID: INTEGER; VAR objName: String8);
32 242 0- A      BEGIN
32 243 --          TRCObjTypeName (objID, String8 (objName));
32 244 -0 A      END { LookupObjName };
32 245 --      { $IFC qTrace } { $D++ } { $ENDC }
32 246 --      { $ENDC qDebug }
32 247 --
32 248 --
32 249 --      { $IFC qDebug }
32 250 --      { $IFC qTrace } { $D+ } { $ENDC }
32 251 --      { $S MAInspect }
32 252 --      {-----+
32 253 --      |      WrLblField              |      Given A5-offset return Object Type Name
32 254 --      +-----+
32 255 -- A      PROCEDURE WrLblField (fieldName: Str255; fieldAddr: Ptr; fieldType: INTEGER);
32 256 0- A      BEGIN
32 257 --          gInspectLinePos := 0;
32 258 --          InspectField (fieldName, fieldAddr, fieldType);
32 259 --          gInspectLinePos := 0;
32 260 -0 A      END { WrLblField };
32 261 --      { $IFC qTrace } { $D++ } { $ENDC }
32 262 --      { $ENDC qDebug }
32 263 --
32 264 --
32 265 --      (*****
32 266 --      (*      T O b j e c t      *)
32 267 --      (*****
32 268 --      { $IFC qDebug }
32 269 --      { $S MAdEbug }
32 270 --      { $IFC qTrace } { $D+ } { $ENDC }
32 271 --      {-----+
32 272 --      |      TypeName                |
32 273 --      +-----+
32 274 -- A      FUNCTION TObjec.TypeName: String8;
32 275 --      VAR
32 276 --          result:      String8;
32 277 0- A      BEGIN
32 278 --          LookupObjName (HInteger (SELF) ^, result);
32 279 --          TypeName := result;
32 280 -0 A      END;
32 281 --      { $IFC qTrace } { $D++ } { $ENDC }
32 282 --
32 283 --
32 284 --      { $IFC qTrace } { $D+ } { $ENDC }
32 285 --      { $S MAFields }
32 286 --      {-----+
32 287 --      |      Fields                  |
32 288 --      +-----+
32 289 -- A      PROCEDURE TObjec.Fields (PROCEDURE DoToField (fieldName: Str255;
32 290 --                                     fieldAddr: Ptr;
32 291 --                                     fieldType: INTEGER));
32 292 0- A      BEGIN
32 293 --          DoToField ('TObjec', NIL, bClass);
32 294 -0 A      END;
32 295 --      { $IFC qTrace } { $D++ } { $ENDC }
32 296 --
32 297 --
32 298 --      { $IFC qTrace } { $D+ } { $ENDC }
32 299 --      { $S MAInspect }
32 300 --      {-----+
32 301 --      |      Inspect                  |
32 302 --      +-----+
32 303 -- A      PROCEDURE TObjec.Inspect;
32 304 --      VAR
32 305 --          hex:      String8;
32 306 0- A      BEGIN
32 307 --          LIntToHex (ORD (SELF), hex, 6);
32 308 --          Writeln (TypeName, '    ', hex);
32 309 --          gInspectLinePos := 0;
32 310 --          Fields (InspectField);
32 311 -0 A      END;
32 312 --      { $IFC qTrace } { $D++ } { $ENDC }
32 313 --      { $ENDC qDebug }
32 314 --
32 315 --
32 316 --      { $IFC qDebug }
32 317 --      { $S MAdEbug }
32 318 --      {-----+
32 319 --      |      IDUobject                |
32 320 --      +-----+
32 321 -- A      PROCEDURE IDUobject;
32 322 0- A      BEGIN
32 323 --          Writeln ('UObject of ', COMPDATE, ' @ ', COMPTIME);
32 324 -0 A      END;
32 325 --      { $ENDC }
32 165 --

```

31 166 -- END.

```

34 1 --      {$P}
34 2 --      { UPatch.p }
34 3 --      { Copyright 1985-1988 by Apple Computer, Inc. All rights reserved. }
34 4 --
34 5 --      { Conditional compilation flags used by this unit:
34 6 --
34 7 --          qNeedsROM128K    True skips code for Mac XL and 64K ROMs,
34 8 --                          false leaves it in.
34 9 --
34 10 --          The settings of $D, $R and $N are imported from UMAUtil.
34 11 --      }
34 12 --
34 13 --      UNIT UPatch;
34 14 --
34 15 --      INTERFACE
34 16 --
34 17 --      {
34 18 --          T H E O R Y   O F   O P E R A T I O N
34 19 --
34 20 --          This unit implements a general trap-patching mechanism. Two types of
34 21 --          patches are implemented:
34 22 --
34 23 --          Normal Patch    The trap is patched such that it jumps to the routine
34 24 --                          of your choice, ignoring the routine it previously
34 25 --                          jumped to. On a 128K ROM machine, the patch is made
34 26 --                          by setting the trap address to your routine. On a 64K
34 27 --                          machine, a small block of code is allocated in the
34 28 --                          system heap. The trap address is set to point to the
34 29 --                          block, and the block contains code to jump to your
34 30 --                          routine.
34 31 --
34 32 --          Tail Patch     The trap is patched such that it jumps to the
34 33 --                          routine of your choice, then jumps to the trap's
34 34 --                          routine. Tail patches come in two flavors: with
34 35 --                          no parameters and with one long-word parameter. The
34 36 --                          patch is made by allocating a small block of code in
34 37 --                          the application heap (system heap for 64K ROMs). The
34 38 --                          trap address is set to the block, and the block contains
34 39 --                          code to jump to your routine and return to the trap's
34 40 --                          original address.
34 41 --
34 42 --          References to patches are kept in TrapPatch records. For each trap
34 43 --          patched you must define a variable of type TrapPatch. This unit links
34 44 --          the TrapPatch records to form a linked list for facilitate unpatching
34 45 --          all patches without specifying each individual patch.
34 46 --      }
34 47 --
34 48 --      {-----+
34 49 --      | Uses                                     |
34 50 --      +-----+
34 51 --      USES
34 52 --          {$LOAD MacIntf.LOAD}
34 53 --          MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
34 54 --          {$LOAD}
34 55 --          UMAUtil;
34 56 --
34 57 --      {-----+
34 58 --      | Global types                             |
34 59 --      +-----+
34 60 --      TYPE
34 61 --          TrapPatchPtr = ^TrapPatch;
34 62 --          TrapPatch =
34 63 --              RECORD
34 64 --              jmpPtr:      Ptr;           { Addr of sys heap block }
34 65 --              trapNum:     INTEGER;       { trap # being patched }
34 66 --              oldTrapAddr: LONGINT;       { old trap address }
34 67 --              nextPatch:   TrapPatchPtr;  { linked list of patches }
34 68 --              END;
34 69 --
34 70 --      {-----+
34 71 --      | Global routines                           |
34 72 --      +-----+
34 73 --      PROCEDURE InitUPatch;
34 74 --          { Call this once to set things up. Initializes the linked list of
34 75 --            patches to NIL. }
34 76 --
34 77 --      FUNCTION PatchTrap(VAR thePatch: TrapPatch; theTrapNum: INTEGER;
34 78 --                          theRoutine: Ptr): INTEGER;
34 79 --          { PatchTrap patches the given trap to theRoutine. On a 128K ROM
34 80 --            machine, we patch the trap's address directly. On a 64K ROM
34 81 --            machine, we must set up a block in the system heap, patch the
34 82 --            trap to jump to the block, and insert code in the block to jump
34 83 --            to theRoutine. }
34 84 --
34 85 --      FUNCTION TailPatch(VAR thePatch: TrapPatch; theTrapNum: INTEGER;
34 86 --                          theRoutine: Ptr): INTEGER;
34 87 --          { TailPatch patches a trap such that it calls theRoutine first, then
34 88 --            executes the old trap routine. theRoutine must be a procedure
34 89 --            with no arguments. This type of patch is implemented by setting
34 90 --            up a block in the system heap, patching the trap to jump to the
34 91 --            block, and inserting code in the block that pushes the original
34 92 --            trap address on the stack and jumps to theRoutine. Thus,
34 93 --            theRoutine returns to the original trap routine. }
34 94 --
34 95 --      FUNCTION Tail1Patch(VAR thePatch: TrapPatch; theTrapNum: INTEGER;
34 96 --                          theRoutine: Ptr): INTEGER;
34 97 --          { Tail1Patch is just like TailPatch, except that it is intended to patch
34 98 --            traps with a single long-word parameter. The parameter passed to

```

```
34 99 --          the trap is also passed to theRoutine. theRoutine must be a
34 100 --          procedure with one long-word parameter. )
34 101 --
34 102 --  PROCEDURE UnpatchTrap(VAR thePatch: TrapPatch);
34 103 --      { UnpatchTrap unpatches the given trap. }
34 104 --
34 105 --  PROCEDURE UnpatchAll;
34 106 --      { UnpatchAll unpatches all traps. }
34 107 --
34 108 --
34 109 --  IMPLEMENTATION
34 110 --
34 111 --  ($I UPatch.incl.p)
```



```

35 1 -- { $P }
35 2 -- { UPatch.incl.p }
35 3 -- { Copyright 1985-1988 by Apple Computer, Inc. All rights reserved. }
35 4 --
35 5 -- {-----+
35 6 -- | Local types |
35 7 -- +-----}
35 8 -- TYPE
35 9 -- { System heap block record for PatchTrap }
35 10 -- PTBlockPtr = ^PTBlock;
35 11 -- PTBlock = RECORD
35 12 --     Jump: INTEGER;
35 13 --     Routine: Ptr;
35 14 -- END;
35 15 --
35 16 -- { System heap block record for TailPatch }
35 17 -- TPBlockPtr = ^TPBlock;
35 18 -- TPBlock = RECORD
35 19 --     MoveL: INTEGER;
35 20 --     OldTrapAddr: LONGINT;
35 21 --     Jump: INTEGER;
35 22 --     Routine: Ptr;
35 23 -- END;
35 24 --
35 25 -- { System heap block record for TailPatch }
35 26 -- TlPBlockPtr = ^TlPBlock;
35 27 -- TlPBlock = RECORD
35 28 --     MoveParameter: LONGINT;
35 29 --     MoveL: INTEGER;
35 30 --     OldTrapAddr: LONGINT;
35 31 --     Jump: INTEGER;
35 32 --     Routine: Ptr;
35 33 -- END;
35 34 --
35 35 --
35 36 -- {-----+
35 37 -- | Local variables |
35 38 -- +-----}
35 39 -- VAR
35 40 --     gPatchList: TrapPatchPtr; { Head of linked list of patches }
35 41 --
35 42 -- (*****
35 43 -- (* Global Routines *)
35 44 -- (*****
35 45 -- { $IFC qTrace } { $D+ } { $ENDC } { Tracing code may not be initialized yet }
35 46 -- { $S MAINInit }
35 47 -- {-----+
35 48 -- | InitUPatch |
35 49 -- +-----}
35 50 -- A PROCEDURE InitUPatch;
35 51 --
35 52 -- A BEGIN
35 53 --     gPatchList := NIL; { Initialize the linked list of patches. }
35 54 -- A END { InitUPatch };
35 55 -- { $IFC qTrace } { $D+ } { $ENDC }
35 56 --
35 57 --
35 58 -- { $IFC qTrace } { $D+ } { $ENDC }
35 59 -- { $S MAINInit }
35 60 -- {-----+
35 61 -- | AllocBlock |
35 62 -- +-----}
35 63 -- A FUNCTION AllocBlock (blockSize: INTEGER; trapNum: INTEGER;
35 64 --     VAR thePatch: TrapPatch): Ptr;
35 65 --
35 66 -- { If blockSize is greater than zero, AllocBlock allocates a non-
35 67 -- relocatable block in the system heap, of size blockSize. It then
35 68 -- fills all fields of thePatch. If blockSize is zero, then no block
35 69 -- is allocated and AllocBlock fills fields of thePatch. }
35 70 --
35 71 -- VAR
35 72 --     theBlock: Ptr;
35 73 --
35 74 -- A BEGIN
35 75 --     IF blockSize > 0 THEN
35 76 --         IF gConfiguration.hasROM128K THEN
35 77 --             theBlock := NewPtr(blockSize)
35 78 --         ELSE
35 79 --             theBlock := NewSysPtr(blockSize)
35 80 --         ELSE
35 81 --             theBlock := NIL;
35 82 --
35 83 --             thePatch.jumpPtr := theBlock;
35 84 --             thePatch.oldTrapAddr := NGetTrapAddress(trapNum, GetTrapType(trapNum));
35 85 --             thePatch.trapNum := trapNum;
35 86 --             thePatch.nextPatch := gPatchList;
35 87 --             gPatchList := @thePatch;
35 88 --             AllocBlock := theBlock;
35 89 -- A END { AllocBlock };
35 90 -- { $IFC qTrace } { $D+ } { $ENDC }
35 91 --
35 92 --
35 93 -- { $IFC qTrace } { $D+ } { $ENDC }
35 94 -- { $S MAINInit }
35 95 -- {-----+
35 96 -- | PatchTrap |
35 97 -- +-----}
35 98 -- A FUNCTION PatchTrap (VAR thePatch: TrapPatch; theTrapNum: INTEGER;

```

```

35 99 --           theRoutine: Ptr): OSErr;
35 100 -- VAR
35 101 --     patchBlock:   PTBlockPtr;
35 102 --
35 103 0- A BEGIN
35 104 --     {$IFC NOT qNeedsROM128K}
35 105 --     IF NOT gConfiguration.hasROM128K THEN
35 106 1- BEGIN
35 107 --
35 108 --         { Allocate the system heap block, and fill thePatch }
35 109 --         patchBlock := PTBlockPtr(AllocBlock(SIZEOF(PTBlock), theTrapNum, thePatch));
35 110 --
35 111 --         IF patchBlock <> NIL THEN
35 112 2- BEGIN
35 113 --             { Stuff the code into the block }
35 114 --             WITH patchBlock^ DO
35 115 3- BEGIN
35 116 --                 Jmp := $4EF9;           { JMP      #Routine }
35 117 --                 Routine := theRoutine;
35 118 -3 END;
35 119 --
35 120 --             { Set the trap address to the block of code in the system heap }
35 121 --             NSetTrapAddress(ORD4(patchBlock), theTrapNum, GetTrapType(theTrapNum));
35 122 -2 END;
35 123 --
35 124 --         PatchTrap := MemError;
35 125 -1 END
35 126 --     ELSE
35 127 --     {$ENDC}
35 128 1- BEGIN
35 129 --         { On 128K ROMs, setup the patch record and the trap address }
35 130 --         patchBlock := PTBlockPtr(AllocBlock(0, theTrapNum, thePatch));
35 131 --         NSetTrapAddress(ORD4(theRoutine), theTrapNum, GetTrapType(theTrapNum));
35 132 --         PatchTrap := noErr;
35 133 -1 END;
35 134 -0 A END { PatchTrap };
35 135 --     {$IFC qTrace}{$D+}{$ENDC}
35 136 --
35 137 --
35 138 --     {$IFC qTrace}{$D+}{$ENDC}
35 139 --     {$S MAINit}
35 140 --     {-----+
35 141 --     |   TailPatch
35 142 --     +-----}
35 143 -- A FUNCTION TailPatch (VAR thePatch: TrapPatch; theTrapNum: INTEGER;
35 144 --                       theRoutine: Ptr): OSErr;
35 145 --
35 146 -- VAR
35 147 --     patchBlock:   TPBlockPtr;
35 148 --
35 149 0- A BEGIN
35 150 --     { Allocate the system heap block, and fill thePatch }
35 151 --     patchBlock := TPBlockPtr(AllocBlock(SIZEOF(TPBlock), theTrapNum, thePatch));
35 152 --
35 153 --     IF patchBlock <> NIL THEN
35 154 1- BEGIN
35 155 --         { Stuff the code into the block }
35 156 --         WITH patchBlock^ DO
35 157 2- BEGIN
35 158 --             MoveL := $2F3C;           { MOVE.L   #OldTrapAddr,-(SP) }
35 159 --             OldTrapAddr := thePatch.OldTrapAddr;
35 160 --             Jmp := $4EF9;           { JMP      #Routine }
35 161 --             Routine := theRoutine;
35 162 -2 END;
35 163 --
35 164 --             { Set the trap address to the block of code in the system heap }
35 165 --             NSetTrapAddress(ORD4(patchBlock), theTrapNum, GetTrapType(theTrapNum));
35 166 -1 END;
35 167 --         TailPatch := MemError;
35 168 -0 A END { TailPatch };
35 169 --     {$IFC qTrace}{$D+}{$ENDC}
35 170 --
35 171 --
35 172 --     {$S MAINit}
35 173 --     {$IFC qTrace}{$D+}{$ENDC}
35 174 --     {-----+
35 175 --     |   TailPatch
35 176 --     +-----}
35 177 -- A FUNCTION TailPatch (VAR thePatch: TrapPatch; theTrapNum: INTEGER;
35 178 --                       theRoutine: Ptr): OSErr;
35 179 --
35 180 -- VAR
35 181 --     patchBlock:   T1PBlockPtr;
35 182 --
35 183 0- A BEGIN
35 184 --     { Allocate the system heap block, and fill thePatch }
35 185 --     patchBlock := T1PBlockPtr(AllocBlock(SIZEOF(T1PBlock), theTrapNum, thePatch));
35 186 --
35 187 --     IF patchBlock <> NIL THEN
35 188 1- BEGIN
35 189 --         { Stuff the code into the block }
35 190 --         WITH patchBlock^ DO
35 191 2- BEGIN
35 192 --             MoveParameter := $2F2F0004; { MOVE.L   4(SP),-(SP) }
35 193 --             MoveL := $2F3C;           { MOVE.L   #OldTrapAddr,-(SP) }
35 194 --             OldTrapAddr := thePatch.OldTrapAddr;
35 195 --             Jmp := $4EF9;           { JMP      #Routine }
35 196 --             Routine := theRoutine;
35 197 -2 END;

```

```

35 198 --
35 199 --      { Set the trap address to the block of code in the system heap }
35 200 --      NSetTrapAddress(ORD4(patchBlock), theTrapNum, GetTrapType(theTrapNum));
35 201 -1      END;
35 202 --      TaillPatch := MemError;
35 203 -0 A  END { TaillPatch };
35 204 --      {IFC qTrace}{SD+}{$ENDC}
35 205 --
35 206 --
35 207 --      {IFC qTrace}{SD+}{$ENDC}
35 208 --      {$S MAterminate}
35 209 --      {-----+
35 210 --      |  UnpatchTrap
35 211 --      +-----}
35 212 -- A  PROCEDURE UnpatchTrap (VAR thePatch: TrapPatch);
35 213 --
35 214 --      VAR
35 215 --          aPatch:      TrapPatchPtr;
35 216 --
35 217 0- A  BEGIN
35 218 --      { Unlink the patch from the linked list of patches }
35 219 --      IF @thePatch = gPatchList THEN
35 220 --          gPatchList := thePatch.nextPatch
35 221 --      ELSE
35 222 1-      BEGIN
35 223 --          aPatch := gPatchList;
35 224 --          WHILE aPatch^.nextPatch <> @thePatch DO
35 225 2-      BEGIN
35 226 --          aPatch := aPatch^.nextPatch;
35 227 --          IF aPatch = NIL THEN
35 228 --              { Couldn't find thePatch, so don't try to unpatch it. }
35 229 --              EXIT(UnpatchTrap);
35 230 -2      END;
35 231 --          aPatch^.nextPatch := thePatch.nextPatch;
35 232 -1      END;
35 233 --
35 234 --      WITH thePatch DO
35 235 1-      BEGIN
35 236 --          { If the patch allocated a block in the system heap, deallocate it }
35 237 --          IF jmpPtr <> NIL THEN
35 238 2-      BEGIN
35 239 --          DisposPtr(jmpPtr);
35 240 --          jmpPtr := NIL;
35 241 -2      END;
35 242 --
35 243 --          { Restore the trap address }
35 244 --          NSetTrapAddress(oldTrapAddr, trapNum, GetTrapType(trapNum));
35 245 -1      END;
35 246 --
35 247 -0 A  END { UnpatchTrap };
35 248 --      {IFC qTrace}{SD+}{$ENDC}
35 249 --
35 250 --
35 251 --      {IFC qTrace}{SD+}{$ENDC}
35 252 --      {$S MAterminate}
35 253 --      {-----+
35 254 --      |  UnpatchAll
35 255 --      +-----}
35 256 -- A  PROCEDURE UnpatchAll;
35 257 --
35 258 0- A  BEGIN
35 259 --      WHILE gPatchList <> NIL DO
35 260 --          UnpatchTrap(gPatchList^);
35 261 -0 A  END { UnpatchAll };
35 262 --      {IFC qTrace}{SD+}{$ENDC}
34 112 --
34 113 --      END.
34 114 --

```



```

36 99 --                                     The default is the entire extent, but
36 100 --                                     it can be changed with the method
36 101 --                                     SetAreaToPrint. )
36 102 --
36 103 -- fFixedSizePages:    ARRAY [VHSelect] OF BOOLEAN;
36 104 --                     { Indicates whether pages are fixed size
36 105 --                     horizontally and vertically. }
36 106 --
36 107 -- fHPrint:             Handle;    { actually of THPrint, a MacPrint type.
36 108 --                                     The Print Record is 120 bytes long,
36 109 --                                     and may be shared among all
36 110 --                                     PrintHandlers in a document or may
36 111 --                                     be unique to each PrintHandler,
36 112 --                                     depending on the flag
36 113 --                                     TDocument.fSharePrintInfo }
36 114 --
36 115 -- fPageStrips:          Point;
36 116 --                     { For printing a view which will take more than one page to print,
36 117 --                     the view is thought of as being divided up by vertical and
36 118 --                     horizontal lines ('page breaks') into a checkerboard of 'pages'.
36 119 --                     A vertical column, one page wide, of such pages is called a
36 120 --                     'vertical page strip', while a horizontal row, one page high,
36 121 --                     of such pages is called a 'horizontal page strip'. The number
36 122 --                     of vertical page strips is represented by fPageStrips.v,
36 123 --                     and the number of horizontal page strips is represented by
36 124 --                     fPageStrips.h
36 125 --
36 126 --                     In a typical Word Processor application, there is only one
36 127 --                     vertical strip of pages, but many horizontal 'strips', each one
36 128 --                     only one page wide. In a graphical application, there may be
36 129 --                     many vertical as well as horizontal strips of pages.
36 130 --
36 131 --                     fPageStrips is recomputed by method CalcPageStrips. )
36 132 --
36 133 -- fStartPage:           INTEGER;    { the page number for the topleftmost
36 134 --                                     corner of the view. Normally 1,
36 135 --                                     but can be set to any positive
36 136 --                                     value }
36 137 --
36 138 -- fPrinterDev:          INTEGER;    { MacPrint's device number for printer;
36 139 --                                     use this at Print time for getting
36 140 --                                     font metrics in device coordinates }
36 141 --
36 142 -- fLastCheckedPrinter:   LONGINT;    { The time that the user's chosen Printer
36 143 --                                     and my own idea of printer metrics
36 144 --                                     were last known to be in harmony }
36 145 --
36 146 -- fLastPrinterName:     STRING[63]; { The name of the current printer driver,
36 147 --                                     at fLastCheckedPrinter. }
36 148 --
36 149 -- fPageDirection:       VHSelect;    { Determines how page numbers are to
36 150 --                                     advance in a large view which is
36 151 --                                     more than one page wide and more
36 152 --                                     than one page high; v means
36 153 --                                     page 2 is BELOW page 1; h means
36 154 --                                     it's to its RIGHT. Irrelevant if
36 155 --                                     the view is only one page wide or
36 156 --                                     only one page high }
36 157 --
36 158 --
36 159 -- fShowBreaks:          BOOLEAN;    { TRUE iff page breaks should be
36 160 --                                     displayed at the moment }
36 161 --
36 162 -- fFinderSetup:         BOOLEAN;    { whether, during Finder Printing, the
36 163 --                                     'Page Setup' dialog should be
36 164 --                                     presented to the user }
36 165 --
36 166 -- fFinderJobDialog:     BOOLEAN;    { whether, during Finder Printing, the
36 167 --                                     Print Job dialog should be
36 168 --                                     presented to the user (making it
36 169 --                                     seem like "Print..." or not
36 170 --                                     (making it seem like "Print One") }
36 171 --
36 172 -- fSquareDots:          BOOLEAN;    { if TRUE, then in the initial instance,
36 173 --                                     use 72 x 72 resolution by default
36 174 --                                     on ImageWriter; if FALSE, then full
36 175 --                                     80 x 72 is used. This flag has no
36 176 --                                     significance if the default printer
36 177 --                                     at the time of creation of the
36 178 --                                     printHandler is not an ImageWriter }
36 179 --
36 180 -- fMinimalMargins:      BOOLEAN;    { If TRUE, page margins are maintained
36 181 --                                     such that exactly the complete
36 182 --                                     printable area of the page is used. }
36 183 --
36 184 -- fLastStrip:           Point;    { The last strip printed. }
36 185 --
36 186 -- fLastBreak:           VPoint;    { The coordinates of the last page break. }
36 187 --
36 188 -- fViewedRect:          VRect;    { The part of the view shown on the
36 189 --                                     "current" page. Set by SetPage. }
36 190 --
36 191 -- { Initialization/Termination }
36 192 -- PROCEDURE TStdPrintHandler.IStdPrintHandler(itsDocument: TDocument; itsView: TView;
36 193 --                                     itsSquareDots, itsHFixedSize, itsVFixedSize: BOOLEAN);
36 194 -- { Initializes the printHandler and installs it in the view; if
36 195 -- 'itsSquareDots' is TRUE, then a default print record for the
36 196 -- ImageWriter world will use square dots ('tall adjusted') rather
36 197 -- than rectangular ones; this sacrifices a bit of resolution but

```

```

36 198 --          ensures that graphics will appear undistorted on the printed
36 199 --          page }
36 200 --
36 201 --      PROCEDURE TStdPrintHandler.Free; OVERRIDE;
36 202 --          { If relevant, frees the PrintInfo record before freeing SELF }
36 203 --
36 204 --      PROCEDURE TStdPrintHandler.Terminate; OVERRIDE;
36 205 --          { Called automatically by MacApp (for the printHandler installed in
36 206 --            the global variable "gPrintHandler" in UMacApp) at the end of
36 207 --            processing to shut down MacPrint code. }
36 208 --
36 209 --      { Page Strips }
36 210 --
36 211 --      PROCEDURE TStdPrintHandler.CalcPageStrips (VAR pageStrips: Point); OVERRIDE;
36 212 --          { Recalculate the number of strips of pages in each dimension }
36 213 --
36 214 --      PROCEDURE TStdPrintHandler.CalcViewPerPage (VAR amtPerPage: VPoint); OVERRIDE;
36 215 --          { Decide how much view (h and v) normally gets mapped into page
36 216 --            interiors }
36 217 --
36 218 --      FUNCTION TStdPrintHandler.BreakFollowing (vhs: VHSelect;
36 219 --          prevBreak: VCoordinate; VAR automatic: BOOLEAN
36 220 --          ): VCoordinate; OVERRIDE;
36 221 --          { Returns the location of the page break which follows the page break
36 222 --            located at 'prevBreak', in direction vhs; returns automatic = TRUE
36 223 --            if the page-break is thought of as an 'automatic' rather than a
36 224 --            'manual' (user-specified) one. Note that page-breaks in dimension
36 225 --            'v' are drawn as vertical lines, those in dimension 'h' as
36 226 --            horizontal lines. Generic support at present is only provided for
36 227 --            'automatic' page breaks; the 'automatic' parameter in
36 228 --            BreakFollowing is a hook to allow clients to support manual page
36 229 --            breaks as well }
36 230 --
36 231 --      PROCEDURE TStdPrintHandler.EachBreak (vhs: VHSelect; includeLast: BOOLEAN;
36 232 --          FUNCTION DoToBreak (loc: VCoordinate; automatic: BOOLEAN): BOOLEAN);
36 233 --          { Iterates through all page-breaks in the dimension given by vhs, or
36 234 --            until DoToBreak returns true; the trailing edge of the view (i.e.,
36 235 --            right or bottom edge) is given as the final break if 'includeLast'
36 236 --            is TRUE. The function 'DoToBreak' is called once for each page break
36 237 --            in the chosen dimension }
36 238 --
36 239 --      PROCEDURE TStdPrintHandler.GetBreakCoord (vhs: VHSelect;
36 240 --          whichBreak: INTEGER; VAR loc: VCoordinate);
36 241 --          { Returns, in 'loc', the coordinate of the 'whichBreak-th' page break
36 242 --            in dimension vhs. For example, GetBreakCoord(v, 5, loc) will return,
36 243 --            in 'loc', the horizontal coordinate of the vertical line
36 244 --            (page-break) which marks the boundary the area of the view which
36 245 --            will be mapped into the 4th vertical strip of pages and the area
36 246 --            of the view which will be mapped into the 5th vertical strip of
36 247 --            pages }
36 248 --
36 249 --      FUNCTION TStdPrintHandler.PageToStrip (pageNumber: INTEGER): Point;
36 250 --          { Returns the horizontal and vertical strip for the given page. }
36 251 --
36 252 --      FUNCTION TStdPrintHandler.StripToPage (hStrip, vStrip: INTEGER): INTEGER;
36 253 --          { Returns the page number for the given page strip. }
36 254 --
36 255 --      FUNCTION TStdPrintHandler.PointToPageStrip (pointInView: VPoint): Point;
36 256 --          { Returns the page strip for a given point in view space. }
36 257 --
36 258 --
36 259 --      { General Utility }
36 260 --
36 261 --      PROCEDURE TStdPrintHandler.GetDocName (VAR docName: Str255);
36 262 --          { Returns the name of the document being printed, for use in
36 263 --            the print dialog. }
36 264 --
36 265 --      PROCEDURE TStdPrintHandler.DoInMacPrint (PROCEDURE WhatToDo);
36 266 --          { If printing code is available, open the Print Driver, then call
36 267 --            procedure "WhatToDo", then close the Print Driver. If printing
36 268 --            code not available, does nothing }
36 269 --
36 270 --      PROCEDURE TStdPrintHandler.OpenPrintShop;
36 271 --          { Attempts to open the print shop by calling PrOpen. Signals
36 272 --            failure if an error occurs. Called from DoInMacPrint. }
36 273 --
36 274 --      PROCEDURE TStdPrintHandler.ClosePrintShop;
36 275 --          { Closes the print shop by calling PrClose. DOES NOT signal
36 276 --            failure if an error occurs, only reports it to the debug
36 277 --            window. }
36 278 --
36 279 --      FUNCTION TStdPrintHandler.MaxPageNumber: INTEGER; OVERRIDE;
36 280 --          { Returns the highest page number which would be printed if the user
36 281 --            requested that "all pages" be printed }
36 282 --
36 283 --
36 284 --      { Actual Printing }
36 285 --          { Basic flow of control at print time:
36 286 --          Print
36 287 --              CheckPrinter
36 288 --              For each page:
36 289 --                  FocusOnBorder
36 290 --                  DrawPageInterior
36 291 --                  FocusOnBorder
36 292 --                  AdornPage }
36 293 --
36 294 --      PROCEDURE TStdPrintHandler.AdornPage;
36 295 --          { Draw things on page # 'aPageNumber' which do not arise from the
36 296 --            View but rather from page considerations; examples: Headers

```

```

36 297 --          and Footers; page numbers; frames drawn around inset text;
36 298 --          row and column headers for spreadsheets. )
36 299 --
36 300 --      PROCEDURE TStdPrintHandler.ChooseSpoolFile(VAR spoolFileName: Str255;
36 301 --          VAR spoolVRefNum: INTEGER; VAR pagesPerSubjob: INTEGER);
36 302 --          { Select the filename and volume to use for spooling; default chooses
36 303 --            empty string on vol 0, which results in MacPrint's automatically
36 304 --            choosing the local Boot volume }
36 305 --
36 306 --      PROCEDURE TStdPrintHandler.DrawPageInterior;
36 307 --          { Make the QuickDraw calls which will result in the 'Interior' of
36 308 --            page # 'aPageNumber' being drawn. The 'Interior' is those
36 309 --            things on the page which arise from drawing the View associated
36 310 --            with the PrintHandler }
36 311 --
36 312 --      PROCEDURE TStdPrintHandler.FocusOnBorder;
36 313 --          { Set up GrafPort, clipping, etc., prior to calling AdornPage }
36 314 --
36 315 --      PROCEDURE TStdPrintHandler.FocusOnInterior; OVERRIDE;
36 316 --          { Set up the GrafPort, clipping, etc., prior to calling
36 317 --            DrawPageInterior }
36 318 --
36 319 --      PROCEDURE TStdPrintHandler.LocatePageInterior(pageNumber: INTEGER;
36 320 --          VAR loc: Point); OVERRIDE;
36 321 --          { Determine where to locate the top-left corner of the Interior of
36 322 --            the given page number }
36 323 --
36 324 --      FUNCTION TStdPrintHandler.OneSubJob(
36 325 --          subjobFirstPage, subjobLastPage: INTEGER;
36 326 --          justSpool: BOOLEAN; partialJob: BOOLEAN;
36 327 --          VAR ranOutOfSpace: BOOLEAN; VAR lastPageTried: INTEGER;
36 328 --          VAR proceed: BOOLEAN): TCommand;
36 329 --          { Processes one subjob for printing. Where appropriate, a print job
36 330 --            is divided automatically into one or more 'subjobs', so that a
36 331 --            long document can be printed even if there is very limited
36 332 --            spool space available on the disk. This is a generalization
36 333 --            of the 'spool-a-page/print-a-page' technique. TStdPrintHandler
36 334 --            will use a subjob size as large as can be handled by current
36 335 --            conditions, and will only fall back to more subjobs, each of
36 336 --            smaller size, if it experiences problems running with a larger
36 337 --            subjob size }
36 338 --
36 339 --      FUNCTION TStdPrintHandler.Print(itsCmdNumber: CmdNumber; VAR proceed: BOOLEAN): TCommand; OVERRIDE;
36 340 --          { Carries out printing for the print handler. }
36 341 --
36 342 --      PROCEDURE TStdPrintHandler.PrintPage(aPageNumber: INTEGER);
36 343 --          { Print the indicated page }
36 344 --
36 345 --      PROCEDURE TStdPrintHandler.SetPage(aPageNumber: INTEGER);
36 346 --          { Dispatches to TView.DoSetInterior and TView.DoSetPageOffset }
36 347 --
36 348 --      PROCEDURE TStdPrintHandler.SetPageInterior(pageNumber: INTEGER); OVERRIDE;
36 349 --          { Responsible for installing the correct values into
36 350 --            fPageAreas.theInterior, representing the 'interior' of the page
36 351 --            in View coordinates }
36 352 --
36 353 --      PROCEDURE TStdPrintHandler.SetPageOffset(coord: VPoint); OVERRIDE;
36 354 --          { Given the view-coordinate of the top-left-most point of the view
36 355 --            which is mapped into the interior of this page, calculate and
36 356 --            set the offset for the print code. }
36 357 --
36 358 --
36 359 --      { Formatting for printing and other Setup methods }
36 360 --
36 361 --      PROCEDURE TStdPrintHandler.CheckPrinter; OVERRIDE;
36 362 --          { Check to see if user has changed printer specifications; if so,
36 363 --            react }
36 364 --
36 365 --      PROCEDURE TStdPrintHandler.InstallMargins(newMargins: Rect;
36 366 --          areMinimalMargins: BOOLEAN);
36 367 --          { Install the (possibly new) margins specified; if
36 368 --            'areMinimalMargins' is TRUE, then the minimal-margins scheme is
36 369 --            used; in this scheme, margins are always maintained at exactly
36 370 --            the size necessary to have the page interior coincide exactly
36 371 --            with the printable area of the page }
36 372 --
36 373 --      PROCEDURE TStdPrintHandler.PrinterChanged; OVERRIDE;
36 374 --          { After printer specifications have changed, absorb the new metrics
36 375 --            and reformat the view if necessary }
36 376 --
36 377 --      PROCEDURE TStdPrintHandler.RedoPageBreaks; OVERRIDE;
36 378 --          { Recalculate the page breaks }
36 379 --
36 380 --      PROCEDURE TStdPrintHandler.Reset; OVERRIDE;
36 381 --          { Reset print record to default }
36 382 --
36 383 --      PROCEDURE TStdPrintHandler.SetDefaultPrintInfo;
36 384 --          { Create a PrintInfo record and initialize it to default values }
36 385 --
36 386 --      PROCEDURE TStdPrintHandler.SetPrintExtent;
36 387 --          { Sets the part of the print handler's view that is printed
36 388 --            by this print handler, by calling the view's GetPrintExtent. }
36 389 --
36 390 --      PROCEDURE TStdPrintHandler.SetMargins;
36 391 --          { Install the desired page margins }
36 392 --
36 393 --      PROCEDURE TStdPrintHandler.ValidatePrintRecord(VAR didChange: BOOLEAN);
36 394 --          { Make certain that the current Print Record is compatible with the
36 395 --            user's current printer choice; 'didChange' is set to TRUE

```

```

36 396 --           iff the kind of printer changed )
36 397 --
36 398 --
36 399 --     { Menu Commands }
36 400 --
36 401 --         FUNCTION TStdPrintHandler.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
36 402 --             { Handles 'show page breaks', among others }
36 403 --
36 404 --         PROCEDURE TStdPrintHandler.DoSetupMenus; OVERRIDE;
36 405 --             { Sets up menus for 'show page breaks' and company }
36 406 --
36 407 --
36 408 --     { Finder Printing }
36 409 --
36 410 --         FUNCTION TStdPrintHandler.SetupForFinder: BOOLEAN; OVERRIDE;
36 411 --             { Sets up the print record for finder printing.  If fFinderPageSetup
36 412 --               is true, then PosePageSetupDialog is called.  If fFinderJobDialog
36 413 --               is true, then PoseJobDialog is called.  Otherwise, prJobMerge is
36 414 --               called to merge the job record from the }
36 415 --
36 416 --         FUNCTION SetupPrintOne: BOOLEAN;
36 417 --             { Sets up the print record for a "print one" type of operation,
36 418 --               one where the entire document or view is printed and no
36 419 --               job dialog takes place.  Returns true if we should proceed. }
36 420 --
36 421 --
36 422 --     { Screen Feedback }
36 423 --
36 424 --         PROCEDURE TStdPrintHandler.DrawPrintFeedback(area: Rect); OVERRIDE;
36 425 --             { Gateway to all screen feedback related to pagination issues }
36 426 --
36 427 --         PROCEDURE TStdPrintHandler.DrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
36 428 --                                                    loc: VCoordinate; automatic: BOOLEAN); OVERRIDE;
36 429 --             { Draws a page break on the screen. }
36 430 --
36 431 --         PROCEDURE TStdPrintHandler.InvalPageFeedback;
36 432 --             { Invalidates (forces redraw) all page-related screen feedback }
36 433 --
36 434 --         FUNCTION TStdPrintHandler.ShowsOnScreen: BOOLEAN;
36 435 --             { Returns TRUE only if the printHandler has an fview which is
36 436 --               in an open window. }
36 437 --
36 438 --
36 439 --     { Dialogs relating to printing }
36 440 --
36 441 --         FUNCTION TStdPrintHandler.PoseJobDialog: BOOLEAN;
36 442 --             { Pose the 'Print Job' dialog }
36 443 --
36 444 --         FUNCTION TStdPrintHandler.PosePageSetupDialog(
36 445 --             VAR proceed: BOOLEAN; isUndoable: BOOLEAN): TCommand;
36 446 --             { Undertake the 'Page Setup' dialog; return a command object to be
36 447 --               performed if relevant }
36 448 --
36 449 --         PROCEDURE TStdPrintHandler.PosePrintDialog;
36 450 --             { Put up the 'Printing in Progress' dialog }
36 451 --
36 452 --         PROCEDURE TStdPrintHandler.ShowDocBeingPrinted (entering: BOOLEAN);
36 453 --             { Put up (or remove) a window whose title is the document's title,
36 454 --               because the Print Manager gets the document title from the
36 455 --               front window. }
36 456 --
36 457 --         PROCEDURE TStdPrintHandler.BanishPrintDialog;
36 458 --
36 459 --     { Debugging }
36 460 --
36 461 --     {IFC.qDebug}
36 462 --         PROCEDURE TStdPrintHandler.IdentifySoftware; OVERRIDE;
36 463 --             { Shows the version of the Printing Unit being used; display is in
36 464 --               the Debug window }
36 465 --
36 466 --         PROCEDURE TStdPrintHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
36 467 --                                                    fieldAddr: Ptr;
36 468 --                                                    fieldType: INTEGER)); OVERRIDE;
36 469 --     {SENDC}
36 470 --         END;
36 471 --
36 472 --
36 473 --     {-----+
36 474 --     | TPrintStyleChangeCommand |
36 475 --     +-----+
36 476 --     TPrintStyleChangeCommand = OBJECT(TCommand)
36 477 --
36 478 --         { This command is used to allow changes made by the user in the
36 479 --           'Page Setup...' dialog to be undoable }
36 480 --
36 481 --     { Fields }
36 482 --
36 483 --         fStdPrintHandler: TStdPrintHandler; { Associated print handler }
36 484 --
36 485 --         fOldHPrint: Handle; { The old print record, before the change }
36 486 --         fNewHPrint: Handle; { The new print record, after the change }
36 487 --
36 488 --
36 489 --     { Initialization and Termination }
36 490 --
36 491 --         PROCEDURE TPrintStyleChangeCommand.IPrintStyleChangeCommand(
36 492 --             itsPrintHandler: TStdPrintHandler);
36 493 --         PROCEDURE TPrintStyleChangeCommand.Free; OVERRIDE;
36 494 --

```



```

36 495 --
36 496 --      { Command Execution }
36 497 --
36 498 --      PROCEDURE TPrintStyleChangeCommand.DoIt; OVERRIDE;
36 499 --      PROCEDURE TPrintStyleChangeCommand.UndoIt; OVERRIDE;
36 500 --      PROCEDURE TPrintStyleChangeCommand.RedoIt; OVERRIDE;
36 501 --      ($IFC qDebug)
36 502 --      { Debugging }
36 503 --      PROCEDURE TPrintStyleChangeCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
36 504 --                                fieldAddr: Ptr;
36 505 --                                fieldType: INTEGER)); OVERRIDE;
36 506 --
36 507 --      {$ENDC}
36 508 --      END;
36 509 --
36 510 --      {-----+
36 511 --      | Global variables |
36 512 --      +-----}
36 513 --      VAR
36 514 --          gBreaksPenState:   PenState; { penState to used to draw page
36 515 --                                     breaks }
36 516 --          gCancelAllPrinting: BOOLEAN; { set to TRUE if during Finder Printing user
36 517 --                                     selects "Cancel All" }
36 518 --          gStdPageMargins:   Rect;     { "Standard" left, top, right, bottom page
36 519 --                                     margins, in screen resolution -
36 520 --                                     one inch all round by default }
36 521 --          gPrintDialog:      DialogPtr; { Used for all printing-related dialogs }
36 522 --
36 523 --          gFinderHPrint:      Handle;   { Used for finder printing, to do PrJobMerge
36 524 --                                     when printing more than one document. }
36 525 --
36 526 --      {-----+
36 527 --      | Global routines |
36 528 --      +-----}
36 529 --      { Initialization }
36 530 --
36 531 --      PROCEDURE InitPrinting;
36 532 --      { Call once, at application start-up time. You can interrogate global
36 533 --        variable 'gCouldPrint' at any time to determine whether printing
36 534 --        code is actually accessible }
36 535 --
36 536 --
36 537 --      { NOTE: None of the remaining global procedures in this unit are intended to
36 538 --        be called directly by the client; they are in the Interface only for
36 539 --        use of TPrintHandler subclasses }
36 540 --
36 541 --      PROCEDURE EmpowerChooser(allowIt: BOOLEAN);
36 542 --      { Set or clear the bit that controls access to the Choose Printer desk
36 543 --        accessory }
36 544 --
36 545 --
36 546 --      IMPLEMENTATION
36 547 --
36 548 --      {$I UPrinting.incl.p}

```

```

37 1 --      {SP}
37 2 --      { UPrinting.incl.p }
37 3 --      { Copyright © 1986-1988 by Apple Computer, Inc. All rights reserved. }
37 4 --
37 5 --      (*
37 6 --          Segmentation strategy:
37 7 --
37 8 --          PrintRes          Resident
37 9 --          PrintMain         Resident and used at initialization
37 10 --          PrintActual      Used only during Actual Printing (imaging/spooling)
37 11 --          PrintImage        Used during imaging only
37 12 --          PrintSpool        Used only during printing of a Spool File
37 13 --          PrintDebug        Debugging code
37 14 --          PrintFinder       Code only ever accessed from Finder Printing
37 15 --          PrintInit         One-time Initialization Code
37 16 --          PrintOpen         Code accessed when opening new document
37 17 --          PrintNonRes        General non-resident code
37 18 --          PrintDoCommand     Code for Page Setup command
37 19 --          PrintTerminate     One-time-only code, called only at Termination time
37 20 --
37 21 --      *)
37 22 --
37 23 --
37 24 --      {-----+
37 25 --      | Local constants          |
37 26 --      +-----}
37 27 --      CONST
37 28 --
37 29 --          kNeverInitialized = MAXINT; { Pseudo-value for fPrinterDev field of TStdPrintHandler; used to
37 30 --                                     trigger an initial PrinterChanged call }
37 31 --
37 32 --          kPrintDriverName = $E000; { STR resource in System file which contains
37 33 --                                     name of the current print driver }
37 34 --
37 35 --      {-----+
37 36 --      | Local variables          |
37 37 --      +-----}
37 38 --      VAR
37 39 --          gPPrPort:          TPPrPort: { Printer Port; used during printing }
37 40 --
37 41 --
37 42 --
37 43 --      (***** Global Routines *****)
37 44 --      (* Global Routines *)
37 45 --      (*****
37 46 --      ($S PrintActual)
37 47 --      {-----+
37 48 --      | CheckButton          |
37 49 --      +-----}
37 50 -- A  PROCEDURE CheckButton;
37 51 --      { This procedure handles events in the 'Print Dialogs' which give the user
37 52 --        'Cancel', and, in the case of printing from the Finder, 'Cancel All Printing. }
37 53 --      CONST
37 54 --          myDlgMask = mDownMask + mUpMask + updateMask;
37 55 --      VAR
37 56 --          box:                Rect;
37 57 --          savedPort:          GrafPtr;
37 58 --          aDialog:            DialogPtr;
37 59 --          anEvent:             EventRecord;
37 60 --          item:                INTEGER;
37 61 --          itemType:            INTEGER;
37 62 --          b:                   BOOLEAN;
37 63 --          printState:          BOOLEAN;
37 64 --          savedPrintHandler:   TPrintHandler;
37 65 -- A  BEGIN
37 66 --      GetPort(savedPort); { Save MacPrint's GrafPort }
37 67 --      IF GetNextEvent(myDlgMask, anEvent) THEN
37 68 --          BEGIN
37 69 --              { Workaround for LaserWriter driver bug: it tries to set GhostWindow
37 70 --                so that its status window is invisible. Unfortunately, it doesn't
37 71 --                always set it, so IsDialogEvent returns FALSE (the status window
37 72 --                is frontmost). If ours isn't already in front, force it there. }
37 73 --              IF gPrintDialog <> FrontWindow THEN
37 74 --                  gApplication.SelectWmgrWindow(gPrintDialog);
37 75 --              IF IsDialogEvent(anEvent) THEN
37 76 --                  BEGIN
37 77 --                      b := DialogSelect(anEvent, aDialog, item);
37 78 --                      IF b AND (aDialog = gPrintDialog) THEN
37 79 --                          CASE item OF
37 80 --                              1: PrSetError(iPrAbort); { Cancel }
37 81 --
37 82 --                              2: BEGIN { Cancel All Printing }
37 83 --                                  PrSetError(iPrAbort);
37 84 --                                  gCancelAllPrinting := TRUE;
37 85 --                                  END;
37 86 --
37 87 --                              END; { case }
37 88 --                  END;
37 89 --              ($IFC FALSE) {??? This doesn't seem to work }
37 90 --              ELSE { not a dialog event - check for updates and activates }
37 91 --                  IF (anEvent.what = updateEvt) OR (anEvent.what = activateEvt) THEN
37 92 --                      BEGIN
37 93 --                          printState := gPrinting;
37 94 --                          gPrinting := FALSE;
37 95 --                          savedPrintHandler := gCurrPrintHandler;
37 96 --                          gCurrPrintHandler := NIL;
37 97 --                          gApplication.HandleEvent(anEvent);
37 98 --                          gPrinting := printState;

```

```

37 99 --          gCurrPrintHandler := savedPrintHandler;
37 100 -2          END;
37 101 --      { $ENDC }
37 102 -1          END;
37 103 --      SetPort(savedPort);      { Restore MacPrint's Grafport }
37 104 -0 A      END;
37 105 --
37 106 --
37 107 --      { $S PrintActual }
37 108 --      {-----+
37 109 --      | ChkPrintErr      |
37 110 --      +-----}
37 111 -- A      PROCEDURE ChkPrintErr(VAR err: OSErr; VAR proceed: BOOLEAN; VAR ranOutOfSpace: BOOLEAN);
37 112 --      { Checks for a MacPrint error, but only if 'proceed' is TRUE on entry; specially
37 113 --      handles one particular error (sets return parameter 'ranOutOfSpace'
37 114 --      if error code indicates disk-space problem while saving a spool file). }
37 115 0- A      BEGIN
37 116 --          IF proceed THEN
37 117 1-              BEGIN
37 118 --                  err := PrError;
37 119 --                  IF err <> noErr THEN
37 120 2-                      BEGIN
37 121 --                          { $IFC qDebug }
37 122 --                          IF gDebugPrinting THEN
37 123 --                              WriteLn('Error from PrError is ', err:1);
37 124 --                          { $ENDC }
37 125 --                          proceed := FALSE;
37 126 --                          IF err = -1 { iPrSavPFile } THEN
37 127 --                              ranOutOfSpace := TRUE;
37 128 -2                          END;
37 129 -1                      END;
37 130 -0 A      END;
37 131 --
37 132 --
37 133 --      { $S PrintMain } { Not the best place, but called at both Init and Term }
37 134 --      {-----+
37 135 --      | EmpowerChooser    |
37 136 --      +-----}
37 137 -- A      PROCEDURE EmpowerChooser(allowIt: BOOLEAN); { Poke low-memory to enable or disable access to 'Choose Printer'
37 138 --                                                    Desk Accessory }
37 139 --
37 140 --      CONST
37 141 --          ChooserByte = $946;
37 142 0- A      BEGIN
37 143 --          IF allowIt THEN
37 144 --              BitSet(Ptr(ChooserByte), 0) { set bit 7 of $946 to 1 to enable, to 0 to disable }
37 145 --          ELSE
37 146 --              BitClr(Ptr(ChooserByte), 0)
37 147 -0 A      END;
37 148 --
37 149 --
37 150 --      { $S PrintRes }
37 151 --      {-----+
37 152 --      | GetDriverName     |
37 153 --      +-----}
37 154 -- A      PROCEDURE GetDriverName (VAR driverName: Str255);
37 155 --      VAR
37 156 --          driverHandle: StringHandle;
37 157 0- A      BEGIN
37 158 --          driverHandle := GetString(kPrintDriverName); { Get current driver }
37 159 --
37 160 --          { Be a little cautious, in case we accidentally pick up something
37 161 --          which is not what we expect, or we can't find it. }
37 162 --          IF (driverHandle <> NIL) & (LENGTH(driverHandle^^) < 64) THEN
37 163 --              driverName := driverHandle^^
37 164 --          ELSE
37 165 --              driverName := '';
37 166 -0 A      END;
37 167 --
37 168 --
37 169 --      { $S PrintInit }
37 170 --      {-----+
37 171 --      | RealInitPrinting  |
37 172 --      +-----}
37 173 -- A      PROCEDURE RealInitPrinting;
37 174 --      VAR
37 175 --          err:          OSErr;
37 176 --          aStdPrintHandler: TStdPrintHandler;
37 177 0- A      BEGIN
37 178 --          gCancelAllPrinting := FALSE;
37 179 --          gPrintDialog := NIL;
37 180 --          gFinderHPrint := NIL;
37 181 --
37 182 --          SetRect(gStdPageMargins, 72, 72, -72, -72); { 1" margins std default }
37 183 --
37 184 --          WITH gBreaksPenState DO
37 185 1-              BEGIN
37 186 --                  pnLoc := Point(0);
37 187 --                  pnSize := Point($00020002);
37 188 --                  pnMode := patCopy;
37 189 --                  StuffHex(@pnPat, 'CC663399CC663399');
37 190 -1              END;
37 191 --
37 192 --          gCouldPrint := TRUE;
37 193 --
37 194 --          IF NOT gChooserOK THEN { disable printer-change in the chooser }
37 195 --              EmpowerChooser(FALSE);
37 196 --
37 197 --          IF gPrintHandler = gNullPrintHandler THEN

```

```

37 198 1-      BEGIN                                { Install a StdPrintHandler in UMacApp global variable gPrintHandler }
37 199 --      New(aStdPrintHandler);
37 200 --      FailNIL(aStdPrintHandler);
37 201 --      aStdPrintHandler.IStdPrintHandler(NIL, NIL, TRUE, TRUE, TRUE);
37 202 --      aStdPrintHandler.fFinderJobDialog := TRUE;
37 203 --      gPrintHandler := aStdPrintHandler;
37 204 -1      END;
37 205 --
37 206 -0 A    END;
37 207 --
37 208 --
37 209 --      {$S Main}
37 210 --      { Initialize the printing unit. Call exactly once, after calling
37 211 --      InitToolBox, and after stuffing gChooserOK to FALSE if you do not wish
37 212 --      to allow your users ever to use 'Choose Printer' while within your
37 213 --      application }
37 214 --      {-----+
37 215 --      |   InitPrinting                               |
37 216 --      +-----}
37 217 -- A    PROCEDURE InitPrinting;
37 218 0- A    BEGIN
37 219 --      RealInitPrinting;
37 220 --      UnloadAllSegments;
37 221 -0 A    END;
37 222 --
37 223 --
37 224 --      {$S PrintSpool}
37 225 --      {-----+
37 226 --      |   PrintSpoolFile                             |
37 227 --      +-----}
37 228 -- A    PROCEDURE PrintSpoolFile(anHPrint: Handle; VAR err: OSErr; VAR proceed: BOOLEAN);
37 229 --      VAR
37 230 --          prStatus:   TPrStatus;
37 231 --          b:          BOOLEAN;
37 232 0- A    BEGIN
37 233 --          proceed := TRUE;
37 234 --          PRPcFile(THPrint(anHPrint), NIL, NIL, NIL, prStatus);
37 235 --          ChkPrintErr(err, proceed, b);
37 236 -0 A    END;
37 237 --
37 238 --      (*****
37 239 --      (*      T S t d P r i n t H a n d l e r      *)
37 240 --      (*****
37 241 --      {$S PrintOpen}
37 242 --      {-----+
37 243 --      |   IStdPrintHandler                           |
37 244 --      +-----}
37 245 -- A    PROCEDURE TStdPrintHandler.IStdPrintHandler(itsDocument: TDocument; itsView: TView;
37 246 --                                                    itsSquareDots, itsHFixedSize, itsVFixedSize: BOOLEAN);
37 247 --      VAR
37 248 --          fi:          FailInfo;
37 249 --
37 250 --          {-----+
37 251 --          |   Hd1InitFailed                             |
37 252 --          +-----}
37 253 -- B    PROCEDURE Hd1InitFailed(error: OSErr; message: LONGINT);
37 254 0- B    BEGIN
37 255 --          Free;
37 256 -0 B    END;
37 257 --
37 258 0- A    BEGIN
37 259 --          fHPrint := NIL;
37 260 --          IPrintHandler(itsView);
37 261 --          fDocument := itsDocument;
37 262 --
37 263 --          CatchFailures(fi, Hd1InitFailed);
37 264 --
37 265 --          fStartPage := 1;
37 266 --          fPageDirection := v;                { Page 2 is below page 1, etc. }
37 267 --          fShowBreaks := FALSE;
37 268 --          fFixedSizePages[h] := itsHFixedSize;
37 269 --          fFixedSizePages[v] := itsVFixedSize;
37 270 --
37 271 --          fPrinterDev := kNeverInitialized;
37 272 --
37 273 --          fFinderSetup := FALSE;
37 274 --          fFinderJobDialog := FALSE;
37 275 --      {$H-}
37 276 --          fSquareDots := itsSquareDots;
37 277 --          SetPt(fLastStrip, MAXINT, MAXINT);
37 278 --          fLastBreak := gZeroVpt;
37 279 --      {$H+}
37 280 --
37 281 --          fPageAreas.theMargins := gStdPageMargins;
37 282 --          fEffectiveDeviceRes.h := 72;
37 283 --          fEffectiveDeviceRes.v := 72;
37 284 --          fMinimalMargins := FALSE;
37 285 --
37 286 --          fLastCheckedPrinter := 0;
37 287 --          IF itsView <> NIL THEN
37 288 1-      BEGIN
37 289 --              { allocates a handle for the print-info, and sets my field fHPrint
37 290 --              accordingly }
37 291 --              SetDefaultPrintInfo;
37 292 --              fView.AttachPrintHandler(SELF);
37 293 -1      END
37 294 --          ELSE
37 295 --              { We leave fView and fHPrint NIL. This must be a global print
37 296 --              handler. }

```

```

37 297 --         itsDocument := NIL;
37 298 --
37 299 --         IF itsDocument <> NIL THEN
37 300 1-             BEGIN
37 301 --                 IF itsDocument.fDocPrintHandler = NIL THEN
37 302 --                     itsDocument.fDocPrintHandler := SELF;
37 303 --                 IF itsDocument.fPrintInfo = NIL THEN
37 304 --                     IF itsDocument.fSharePrintInfo THEN
37 305 --                         itsDocument.fPrintInfo := fHPrint;
37 306 -1             END;
37 307 --
37 308 --         Success(fi);
37 309 -0 A     END;
37 310 --
37 311 --
37 312 --         {$S PrintClose}
37 313 --         {-----+
37 314 --         |   Free
37 315 --         +-----}
37 316 -- A     PROCEDURE TStdPrintHandler.Free; OVERRIDE;
37 317 --     VAR
37 318 --         dontDispose:    BOOLEAN;
37 319 --         itsDocument:    TDocument;
37 320 0- A     BEGIN
37 321 --         dontDispose := fView <> NIL;
37 322 --         IF dontDispose THEN
37 323 1-             BEGIN
37 324 --                 IF fView.fPrintHandler = SELF THEN
37 325 --                     fView.AttachPrintHandler(qNullPrintHandler);
37 326 --                 itsDocument := fDocument;
37 327 --                 dontDispose := itsDocument <> NIL;
37 328 --                 IF dontDispose THEN
37 329 2-                     BEGIN
37 330 --                         IF itsDocument.fDocPrintHandler = SELF THEN
37 331 --                             itsDocument.fDocPrintHandler := NIL;
37 332 --                         dontDispose := itsDocument.fSharePrintInfo;
37 333 -2                     END;
37 334 --                 IF dontDispose THEN
37 335 --                     dontDispose := itsDocument.fPrintInfo = fHPrint;
37 336 -1             END;
37 337 --         IF NOT dontDispose THEN
37 338 --             IF fHPrint <> NIL THEN
37 339 --                 DisposHandle(Handle(fHPrint));
37 340 --             INHERITED Free;
37 341 -0 A     END;
37 342 --
37 343 --
37 344 --         {$S PrintImage}
37 345 --         {-----+
37 346 --         |   AdornPage
37 347 --         +-----}
37 348 -- A     PROCEDURE TStdPrintHandler.AdornPage;
37 349 --     ($IFC qDebug)
37 350 --     VAR
37 351 --         heading:    Str255;
37 352 --         pgStr:      Str255;
37 353 --         handyRect:  Rect;
37 354 --         xLoc:       INTEGER;
37 355 --         rPlusL:     INTEGER;
37 356 --     ($ENDC)
37 357 0- A     BEGIN
37 358 --     ($IFC qDebug)
37 359 --         IF qDebugPrinting THEN          { Print extra stuff if debugging }
37 360 1-             BEGIN
37 361 --                 NumToString(fFocusedPage, pgStr);
37 362 --                 TextFont(applFont);
37 363 --                 TextFace([]);
37 364 --                 TextSize(12);
37 365 --                 heading := CONCAT('-', pgStr, '-'); { ??? Make easier for client to change this }
37 366 --
37 367 --                 { avoid a $H problem by precomputing the values we need from fPageAreas.thePaper }
37 368 --                 WITH fPageAreas.thePaper DO
37 369 --                     rPlusL := right + left;
37 370 --
37 371 --                 xLoc := (rPlusL - StringWidth(heading)) DIV 2;
37 372 --                 MoveTo(xLoc, fPageAreas.theInk.bottom - 8); { ??? Arbitrary choice }
37 373 --                 DrawString(heading);
37 374 --
37 375 --                 { Additionally frame the printable area of the page if qDebugPrinting }
37 376 --                 handyRect := fPageAreas.theInk;
37 377 --                 PenSize(1, 1);
37 378 --                 FrameRect(handyRect);
37 379 --
37 380 --                 { Frame the 'interior' of the page }
37 381 --                 PenSize(2, 2);
37 382 --                 handyRect := fPageAreas.theInterior;
37 383 --                 FrameRect(handyRect);
37 384 -1             END;
37 385 --     ($ENDC)
37 386 -0 A     END;
37 387 --
37 388 --
37 389 --         {$S PrintActual}
37 390 --         {-----+
37 391 --         |   BanishPrintDialog
37 392 --         +-----}
37 393 -- A     PROCEDURE TStdPrintHandler.BanishPrintDialog;
37 394 0- A     BEGIN
37 395 --         IF qPrintDialog <> NIL THEN

```

```

37 396 1-      BEGIN
37 397 --      DisposDialog(gPrintDialog);
37 398 --      gPrintDialog := NIL;
37 399 -1      END;
37 400 -0 A  END;
37 401 --
37 402 --
37 403 --      {$S PrintRes}
37 404 --      {-----+
37 405 --      | BreakFollowing |
37 406 --      +-----}
37 407 -- A  FUNCTION TStdPrintHandler.BreakFollowing(vhs: VHSelect; prevBreak: VCoordinate;
37 408 --      VAR automatic: BOOLEAN): VCoordinate; OVERRIDE;
37 409 --      { Called from fView.DoBreakFollowing, }
37 410 --
37 411 --      VAR
37 412 --      orthoVHS:  VHSelect;
37 413 --      newLoc:    VCoordinate;
37 414 0- A  BEGIN
37 415 --      orthoVHS := gOrthogonal[vhs];
37 416 --      automatic := TRUE;
37 417 --      newLoc := Min(prevBreak + fViewPerPage.vh[orthoVHS], fPrintExtent.botRight.vh[orthoVHS]);
37 418 --
37 419 --      {$IFC qDebug}
37 420 --      IF newLoc <= prevBreak THEN
37 421 1-      BEGIN
37 422 --      WriteLn('No advance in BreakFollowing: vhs = ', ORD(vhs):1, ' prevBreak = ', prevBreak:1,
37 423 --      ' newLoc = ', newLoc:1, ' view size: ', fPrintExtent.botRight.vh[orthoVHS]:1);
37 424 --      ProgramBreak('No advance in BreakFollowing');
37 425 -1      END;
37 426 --      {$ENDC}
37 427 --
37 428 --      BreakFollowing := newLoc;
37 429 -0 A  END;
37 430 --
37 431 --
37 432 --      {$S PrintNonRes}
37 433 --      {-----+
37 434 --      | CalcPageStrips |
37 435 --      +-----}
37 436 -- A  PROCEDURE TStdPrintHandler.CalcPageStrips (VAR pageStrips: Point); OVERRIDE;
37 437 --      VAR
37 438 --      vhs,
37 439 --      ortho:  VHSelect;
37 440 --      nStrips: INTEGER;
37 441 --
37 442 --      {-----+
37 443 --      | FindLimit |
37 444 --      +-----}
37 445 -- B  FUNCTION FindLimit(loc: VCoordinate; automatic: BOOLEAN): BOOLEAN;
37 446 0- B  BEGIN
37 447 --      nStrips := nStrips + 1;
37 448 --      FindLimit := FALSE;
37 449 -0 B  END;
37 450 --
37 451 0- A  BEGIN
37 452 --
37 453 --      { If pages are of fixed size, then simple divide the total size by the
37 454 --      page size. Otherwise, count up the page breaks one by one. }
37 455 --
37 456 --      FOR vhs := v TO h DO
37 457 1-      BEGIN
37 458 --      ortho := gOrthogonal[vhs];
37 459 --      IF fFixedSizePages[ortho] THEN
37 460 --      ($H-)
37 461 --      WITH fPrintExtent, fViewPerPage DO
37 462 --      pageStrips.vh[vhs] :=
37 463 --      (botRight.vh[ortho] - topLeft.vh[ortho] + vh[ortho] - 1) DIV vh[ortho]
37 464 --      ($H+)
37 465 --      ELSE
37 466 2-      BEGIN
37 467 --      nStrips := 0;
37 468 --      EachBreak(vhs, TRUE, FindLimit);
37 469 --      pageStrips.vh[vhs] := nStrips;
37 470 -2      END;
37 471 -1      END;
37 472 -0 A  END;
37 473 --
37 474 --
37 475 --      {$S PrintNonRes}
37 476 --      {-----+
37 477 --      | CalcViewPerPage |
37 478 --      +-----}
37 479 -- A  PROCEDURE TStdPrintHandler.CalcViewPerPage(VAR amtPerPage: VPoint); OVERRIDE;
37 480 --      VAR
37 481 --      vhs:  VHSelect;
37 482 0- A  BEGIN
37 483 --      WITH fPageAreas DO ($H-)
37 484 --      FOR vhs := v TO h DO
37 485 --      amtPerPage.vh[vhs] := MAX(1,
37 486 --      ((ORD4(thePaper.botRight.vh[vhs] - thePaper.topLeft.vh[vhs]
37 487 --      - ABS(theMargins.topLeft.vh[vhs])
37 488 --      - ABS(theMargins.botRight.vh[vhs])))); ($H+)
37 489 -0 A  END;
37 490 --
37 491 --
37 492 --      {$S PrintRes}
37 493 --      {-----+
37 494 --      | CheckPrinter |

```

```

37 495 -- +-----+
37 496 -- A  PROCEDURE TStdPrintHandler.CheckPrinter; OVERRIDE;
37 497 -- VAR
37 498 --     oldPageAreas:   PageAreas;
37 499 --     oldPrinterDev:  INTEGER;
37 500 --     oldDevRes:      Point;
37 501 --     oldEffectiveRes: Point;
37 502 --     didChange:      BOOLEAN;
37 503 --     msgSent:        BOOLEAN;
37 504 --     driverName:     Str255;
37 505 --
37 506 --     {-----+
37 507 --     | ItChanged |
37 508 --     +-----+
37 509 -- B  PROCEDURE ItChanged(aView: TView); { this msg is sent to all views if my associated document is the
37 510 --                                         kind where one copy of the PrintInfo is shared among all views }
37 511 0- B  BEGIN
37 512 --     aView.DoPrinterChanged;
37 513 -0 B  END;
37 514 0- A  BEGIN
37 515 --     oldPageAreas := fPageAreas;
37 516 --     oldPrinterDev := fPrinterDev;
37 517 --     oldDevRes := fDeviceRes;
37 518 --     oldEffectiveRes := fEffectiveDeviceRes;
37 519 --
37 520 --     { It costs an open and several LoadResources to call PrValidate. Before
37 521 --     doing so, check to see if the printer has actually changed. }
37 522 --     IF fLastCheckedPrinter < gLastDeskAcc THEN
37 523 --     { If we haven't checked printer since the last time
37 524 --     the user might have run the Chooser... }
37 525 1- BEGIN
37 526 --     GetDriverName(driverName);
37 527 --     IF NOT EqualString(fLastPrinterName, driverName, FALSE, TRUE) THEN
37 528 2- BEGIN
37 529 --     { Printer name has changed; remember the new one and re-validate
37 530 --     the print record. }
37 531 --     BlockMove(@driverName, @fLastPrinterName, LENGTH(driverName));
37 532 --     ValidatePrintRecord(didChange);
37 533 -2 END;
37 534 --     fLastCheckedPrinter := TickCount;
37 535 -1 END;
37 536 --
37 537 --     WITH THPrint(fHPrint)^ DO
37 538 1- ($H-) BEGIN
37 539 --     fPageAreas.thePaper := rPaper;
37 540 --
37 541 --     WITH fPageAreas.thePaper, fEffectiveDeviceRes DO { Recompute effective device resolution }
37 542 2- BEGIN
37 543 --     h := (IntMultiply(iPrPgFract, right - left)) DIV prStl.iPageH;
37 544 --     v := (IntMultiply(iPrPgFract, bottom - top)) DIV prStl.iPageV;
37 545 -2 END;
37 546 --
37 547 --     WITH prInfo DO { Store latest data from MacPrint prInfo record into my own instance variables }
37 548 2- BEGIN
37 549 --     fPrinterDev := iDev;
37 550 --     SetPt(fDeviceRes, iHRes, iVRes);
37 551 --     WITH fPageAreas DO
37 552 3- BEGIN
37 553 --     theInk := rPage;
37 554 --     IF NOT fMinimalMargins THEN
37 555 --     SetRect(theMargins, IntMultiply(theMargins.left, fEffectiveDeviceRes.h) DIV oldEffectiveRes.h,
37 556 --     IntMultiply(theMargins.top, fEffectiveDeviceRes.v) DIV oldEffectiveRes.v,
37 557 --     IntMultiply(theMargins.right, fEffectiveDeviceRes.h) DIV oldEffectiveRes.h,
37 558 --     IntMultiply(theMargins.bottom, fEffectiveDeviceRes.v) DIV oldEffectiveRes.v);
37 559 -3 END;
37 560 -2 END;
37 561 --     ($H+)
37 562 -1 END;
37 563 --
37 564 --     IF (NOT EqualRect(fPageAreas.thePaper, oldPageAreas.thePaper)) OR
37 565 --     (NOT EqualRect(fPageAreas.theInk, oldPageAreas.theInk)) OR
37 566 --     (NOT EqualPt(fDeviceRes, oldDevRes)) OR
37 567 --     (oldPrinterDev = kNeverInitialized) THEN
37 568 1- BEGIN { something important to our projection model changed... }
37 569 --     msgSent := FALSE;
37 570 --     IF fDocument <> NIL THEN
37 571 --     IF fDocument.fSharePrintInfo THEN
37 572 2- BEGIN
37 573 --     fDocument.ForAllViewsDo(ItChanged);
37 574 --     msgSent := TRUE;
37 575 -2 END;
37 576 --
37 577 --     IF NOT msgSent THEN { didn't send msg to all views, current company included, so
37 578 --     now we need to tell must my local view }
37 579 --     fView.DoPrinterChanged;
37 580 -1 END;
37 581 -0 A  END;
37 582 --
37 583 --
37 584 --     {$S PrintImage}
37 585 --     {-----+
37 586 --     | ChooseSpoolFile |
37 587 --     +-----+
37 588 -- A  PROCEDURE TStdPrintHandler.ChooseSpoolFile(VAR spoolFileName: Str255; VAR spoolVRefNum: INTEGER;
37 589 --     VAR pagesPerSubjob: INTEGER);
37 590 0- A  BEGIN
37 591 --     spoolFileName := ''; { Default choices tell MacPrint to use its standard choice algorithm }
37 592 --     spoolVRefNum := 0;
37 593 --     { Print Shop suggests attempting to print entire document. If it

```

```

37 594 --      runs out of space, then PerformPrinting will retry with a
37 595 --      smaller number of pages. )
37 596 --      pagesPerSubjob := MAXINT;
37 597 -0 A  END;
37 598 --
37 599 --
37 600 --      {$S PrintRes}
37 601 --      {-----+
37 602 --      |   ClosePrintShop   |
37 603 --      +-----+}
37 604 -- A  PROCEDURE TStdPrintHandler.ClosePrintShop;
37 605 --
37 606 --      VAR
37 607 --          err:      OSErr;
37 608 --
37 609 0- A  BEGIN
37 610 --          PrClose;
37 611 --          {$IFC qDebug}
37 612 --              err := PrError;      { Now check for fresh error from MacPrint -- only for debugging, since
37 613 --                                  ChkPrintErr checks afresh in the real world }
37 614 --              IF err <> noErr THEN
37 615 --                  WriteLn('Error from MacPrint is: ', err:1);
37 616 --          {$ENDC}
37 617 -0 A  END;
37 618 --
37 619 --
37 620 --      {$S PrintRes}
37 621 --      {-----+
37 622 --      |   DoInMacPrint   |
37 623 --      +-----+}
37 624 -- A  PROCEDURE TStdPrintHandler.DoInMacPrint(PROCEDURE WhatToDo);
37 625 --      VAR
37 626 --          fi:      FailInfo;
37 627 --
37 628 --      {-----+
37 629 --      |   Hd1PrintFailure   |
37 630 --      +-----+}
37 631 -- B  PROCEDURE Hd1PrintFailure(error: OSErr; message: LONGINT);
37 632 0- B  BEGIN
37 633 --          ClosePrintShop;
37 634 -0 B  END;
37 635 --
37 636 0- A  BEGIN
37 637 --      IF gCouldPrint THEN
37 638 1-      BEGIN
37 639 --          PrSetError(noErr);      { Clear printer-error flag }
37 640 --          CatchFailures(fi, Hd1PrintFailure);
37 641 --          OpenPrintShop;
37 642 --          WhatToDo;      { Do what needs to be done }
37 643 --          Success(fi);
37 644 --          ClosePrintShop;
37 645 -1      END;
37 646 --      { NB: if gCouldPrint is FALSE, DoInMacPrint is a no-op }
37 647 -0 A  END;
37 648 --
37 649 --
37 650 --      {$S PrintSelCommand}
37 651 --      {-----+
37 652 --      |   DoMenuCommand   |
37 653 --      +-----+}
37 654 -- A  FUNCTION TStdPrintHandler.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
37 655 --      VAR
37 656 --          proceed:      BOOLEAN;
37 657 --
37 658 0- A  BEGIN
37 659 --          DoMenuCommand := gNoChanges;
37 660 1-      CASE aCmdNumber OF
37 661 --
37 662 --          cPrint:
37 663 2-          BEGIN
37 664 --              CheckPrinter;
37 665 --              IF PoseJobDialog THEN
37 666 --                  DoMenuCommand := Print(aCmdNumber, proceed);
37 667 -2          END;
37 668 --          cPrintOne:
37 669 2-          BEGIN
37 670 --              CheckPrinter;
37 671 --              IF SetupPrintOne THEN
37 672 --                  DoMenuCommand := Print(aCmdNumber, proceed);
37 673 -2          END;
37 674 --          cPageSetup:
37 675 --              DoMenuCommand := PosePageSetupDialog(proceed, TRUE);
37 676 --
37 677 --          cShowBreaks:      { Toggle state of "Show Breaks" }
37 678 2-          BEGIN
37 679 --              fShowBreaks := NOT fShowBreaks;
37 680 --              InvalPageFeedback; { force redraw of area the breaks did or will occupy }
37 681 -2          END;
37 682 --
37 683 --          OTHERWISE
37 684 --              DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
37 685 -1          END { CASE };
37 686 -0 A  END;
37 687 --
37 688 --
37 689 --      {$S PrintRes}
37 690 --      {-----+
37 691 --      |   DoSetupMenus   |
37 692 --      +-----+}

```



```

37 693 -- A  PROCEDURE TStdPrintHandler.DoSetupMenus; OVERRIDE;
37 694 0- A  BEGIN
37 695 --      INHERITED DoSetupMenus;
37 696 --
37 697 --      IF gCouldPrint AND (fView <> NIL) THEN
37 698 1-          BEGIN
37 699 --              Enable(cPrint, TRUE);
37 700 --              Enable(cPageSetup, TRUE);
37 701 --              Enable(cPrintOne, TRUE);
37 702 -1          END;
37 703 --
37 704 --              EnableCheck(cShowBreaks, TRUE, fShowBreaks);
37 705 -0 A  END;
37 706 --
37 707 --
37 708 --      {$S PrintRes}
37 709 --      {-----+
37 710 --      | DrawPrintFeedback
37 711 --      +-----}
37 712 -- A  PROCEDURE TStdPrintHandler.DrawPrintFeedback(area: Rect); OVERRIDE;
37 713 --      ( Draws page breaks and page numbers )
37 714 --      VAR vhs:          VHSelect;
37 715 --          orthovhs:    VHSelect;
37 716 --          whichBreak:   INTEGER;
37 717 --          viewArea:     VRect;
37 718 --      {$IFC qDebug}
37 719 --          pageNumStyle:  TextStyle;
37 720 --      {$ENDC}
37 721 --
37 722 --      {-----+
37 723 --      | DrawABreak
37 724 --      +-----}
37 725 -- B  FUNCTION DrawABreak (loc: VCoordinate; automatic: BOOLEAN): BOOLEAN;
37 726 --
37 727 0- B  BEGIN
37 728 --      IF loc > viewArea.botRight.vh[orthovhs] THEN
37 729 --          DrawABreak := TRUE
37 730 --      ELSE
37 731 1-          BEGIN
37 732 --              DrawABreak := FALSE;
37 733 --              whichBreak := whichBreak + 1;
37 734 --              IF (loc > viewArea.topLeft.vh[orthovhs] - gBreaksPenState.pnSize.vh[orthovhs]) THEN
37 735 --                  fView.DoDrawPageBreak(vhs, whichBreak, loc, automatic);
37 736 -1              END;
37 737 -0 B  END;
37 738 --
37 739 0- A  BEGIN
37 740 --      IF fShowBreaks OR gDebugPrinting THEN
37 741 1-          BEGIN
37 742 --              SetPrintExtent; { Make sure print extent is accurate before starting }
37 743 --
37 744 --      {$IFC qDebug}
37 745 --          IF gDebugPrinting THEN { Now draw Page numbers in the corners of pages, if desired }
37 746 2-          BEGIN
37 747 --              SetTextStyle(pageNumStyle, ApplFont, [bold], 9, gRGBBlack);
37 748 --              SetPortTextStyle(pageNumStyle);
37 749 -2              END;
37 750 --      {$ENDC}
37 751 --
37 752 --              SetPenState(gBreaksPenState);
37 753 --              fView.QDToViewRect(area, viewArea);
37 754 --
37 755 --              FOR vhs := v TO h DO
37 756 2-                  BEGIN
37 757 --                      orthovhs := gOrthogonal[vhs];
37 758 --                      whichBreak := 0;
37 759 --                      EachBreak(vhs, FALSE, DrawABreak);
37 760 -2                  END;
37 761 -1              END;
37 762 -0 A  END;
37 763 --
37 764 --
37 765 --      {$S PrintRes}
37 766 --      {-----+
37 767 --      | DrawPageBreak
37 768 --      +-----}
37 769 -- A  PROCEDURE TStdPrintHandler.DrawPageBreak (vhs: VHSelect; whichBreak: INTEGER;
37 770 --                                              loc: VCoordinate; automatic: BOOLEAN); OVERRIDE;
37 771 --
37 772 --      VAR
37 773 --          vPt:          VPoint;
37 774 --          qdStartPt:    Point;
37 775 --          qdEndPt:      Point;
37 776 --      {$IFC qDebug}
37 777 --          i:            INTEGER;
37 778 --          hLoc:          VCoordinate;
37 779 --          aString:       Str255;
37 780 --      {$ENDC}
37 781 --
37 782 0- A  BEGIN
37 783 --          vPt.vh[gOrthogonal[vhs]] := loc;
37 784 --          vPt.vh[vhs] := 0;
37 785 --          qdStartPt := fView.ViewToQDPt(vPt);
37 786 --          vPt.vh[vhs] := fView.fSize.vh[vhs] - gBreaksPenState.pnSize.vh[vhs];
37 787 --          qdEndPt := fView.ViewToQDPt(vPt);
37 788 --
37 789 --          IF fShowBreaks THEN
37 790 1-              BEGIN
37 791 --                  MoveTo(qdStartPt.h, qdStartPt.v);

```

```

37 792 --      LineTo(qdEndPt.h, qdEndPt.v);
37 793 --      END;
37 794 --
37 795 --      {IFC qDebug}
37 796 --      IF qDebugPrinting THEN
37 797 --      IF vhs = h THEN
37 798 --          FOR i := 0 TO fPageStrips.v DO
37 799 --          BEGIN
37 800 --              IF i = 0 THEN
37 801 --                  hLoc := 0
37 802 --              ELSE
37 803 --                  GetBreakCoord(v, i, hLoc);
37 804 --
37 805 --                  NumToString(StripToPage(whichBreak - 1, i), aString);
37 806 --                  MoveTo(hLoc + 3, qdStartPt.v - 3);
37 807 --                  DrawString(aString);
37 808 --
37 809 --                  NumToString(StripToPage(whichBreak, i), aString);
37 810 --                  MoveTo(hLoc + 3, qdEndPt.v + 10);
37 811 --                  DrawString(aString);
37 812 --
37 813 --                  NumToString(StripToPage(whichBreak - 1, i - 1), aString);
37 814 --                  MoveTo(hLoc - StringWidth(aString) - 3, qdStartPt.v - 3);
37 815 --                  DrawString(aString);
37 816 --
37 817 --                  NumToString(StripToPage(whichBreak, i - 1), aString);
37 818 --                  MoveTo(hLoc - StringWidth(aString) - 3, qdEndPt.v + 10);
37 819 --                  DrawString(aString);
37 820 --              END;
37 821 --          {$ENDC}
37 822 --      END;
37 823 --
37 824 --
37 825 --      {$$ PrintImage}
37 826 --      {-----+
37 827 --      | DrawPageInterior
37 828 --      +-----}
37 829 -- A  PROCEDURE TStdPrintHandler.DrawPageInterior;
37 830 -- A  BEGIN
37 831 --      fView.DrawContents; { i.e., by default, the same code used for drawing on the screen }
37 832 -- A  END;
37 833 --
37 834 --
37 835 --      {$$ PrintRes}
37 836 --      {-----+
37 837 --      | EachBreak
37 838 --      +-----}
37 839 -- A  PROCEDURE TStdPrintHandler.EachBreak(vhs: VHSelct; includeLast: BOOLEAN;
37 840 --      FUNCTION DoToBreak(loc: VCoordinate; automatic: BOOLEAN): BOOLEAN);
37 841 --      VAR
37 842 --          startLoc: VCoordinate;
37 843 --          endLoc: VCoordinate;
37 844 --          loc: VCoordinate;
37 845 --          automatic: BOOLEAN;
37 846 --          done: BOOLEAN;
37 847 -- A  BEGIN
37 848 --      WITH fPrintExtent DO
37 849 --      BEGIN
37 850 --          startLoc := topLeft.vh[gOrthogonal[vhs]];
37 851 --          endLoc := botRight.vh[gOrthogonal[vhs]];
37 852 --      END;
37 853 --
37 854 --      loc := startLoc;
37 855 --      automatic := TRUE;
37 856 --      done := FALSE;
37 857 --      WHILE (loc < endLoc) AND NOT done DO
37 858 --      BEGIN
37 859 --          IF loc <> startLoc THEN
37 860 --              done := DoToBreak(loc, automatic);
37 861 --          loc := fView.DoBreakFollowing(vhs, loc, automatic);
37 862 --      END;
37 863 --
37 864 --      IF includeLast THEN
37 865 --          done := DoToBreak(loc, automatic);
37 866 -- A  END;
37 867 --
37 868 --
37 869 --      {IFC qDebug}
37 870 --      {$$ PrintFields}
37 871 --      {-----+
37 872 --      | Fields
37 873 --      +-----}
37 874 -- A  PROCEDURE TStdPrintHandler.Fields (PROCEDURE DoToField (fieldName: Str255;
37 875 --      fieldAddr: Ptr;
37 876 --      fieldType: INTEGER));
37 877 -- A  BEGIN
37 878 --      DoToField('TStdPrintHandler', NIL, bClass);
37 879 --      DoToField('fPageAreas', NIL, bTitle);
37 880 --      DoToField(' thePaper', @fPageAreas.thePaper, bRect);
37 881 --      DoToField(' theInk', @fPageAreas.theInk, bRect);
37 882 --      DoToField(' theMargins', @fPageAreas.theMargins, bRect);
37 883 --      DoToField(' theInterior', @fPageAreas.theInterior, bRect);
37 884 --      DoToField('fDocument', @fDocument, bObject);
37 885 --      DoToField('fPrintExtent', @fPrintExtent, bVRect);
37 886 --      DoToField('fFixedSize[h]', @fFixedSizePages[h], bBoolean);
37 887 --      DoToField('fFixedSize[v]', @fFixedSizePages[v], bBoolean);
37 888 --      DoToField('fHPrint', @fHPrint, bHandle);
37 889 --      DoToField('fPageStrips', @fPageStrips, bPoint);
37 890 --      DoToField('fStartPage', @fStartPage, bInteger);

```

```

37 891 --      DoToField('fPrinterDev', @fPrinterDev, bInteger);
37 892 --      DoToField('fLastCheckedPrinter', @fLastCheckedPrinter, bLongInt);
37 893 --      DoToField('fLastPrinterName', @fLastPrinterName, bString);
37 894 --      DoToField('fPageDirection', @fPageDirection, bByte);
37 895 --      DoToField('fShowBreaks', @fShowBreaks, bBoolean);
37 896 --      DoToField('fFinderSetup', @fFinderSetup, bBoolean);
37 897 --      DoToField('fFinderJobDialog', @fFinderJobDialog, bBoolean);
37 898 --      DoToField('fSquareDots', @fSquareDots, bBoolean);
37 899 --      DoToField('fMinimalMargins', @fMinimalMargins, bBoolean);
37 900 --      DoToField('fLastStrip', @fLastStrip, bPoint);
37 901 --      DoToField('fLastBreak', @fLastBreak, bVPoint);
37 902 --      DoToField('fViewedRect', @fViewedRect, bVRect);
37 903 --      INHERITED Fields(DoToField);
37 904 -0 A  END;
37 905 --      { $ENDC }
37 906 --
37 907 --
37 908 --      { $S PrintImage }
37 909 --      {-----+
37 910 --      |   FocusOnBorder
37 911 --      +-----+}
37 912 -- A  PROCEDURE TStdPrintHandler.FocusOnBorder;
37 913 --      VAR
37 914 --          rectToClipTo: Rect;
37 915 0- A  BEGIN
37 916 --          rectToClipTo := fPageAreas.theInk;
37 917 --          WITH fPageAreas.theInk DO
37 918 --              { $H- }
37 919 --                  SetOrigin(left, top);          { Only works for newer LaserWriter drivers }
37 920 --              { $H+ }
37 921 --              ClipRect(rectToClipTo);
37 922 -0 A  END;
37 923 --
37 924 --
37 925 --      { $S PrintImage }
37 926 --      {-----+
37 927 --      |   FocusOnInterior
37 928 --      +-----+}
37 929 -- A  PROCEDURE TStdPrintHandler.FocusOnInterior; OVERRIDE;
37 930 --      VAR
37 931 --          rectToClip:      Rect;
37 932 --          aRect:           Rect;
37 933 --          theOrigin:       Point;
37 934 --          vhs:             VHSelect;
37 935 0- A  BEGIN
37 936 --      { $H- }
37 937 --          WITH fPageAreas DO
37 938 --              BEGIN
37 939 --                  aRect := theInk;
37 940 --                  theOrigin := theInk.topLeft;
37 941 --                  FOR vhs := v TO h DO
37 942 --                      IF fView.fsize.vh[vhs] > kMaxCoord THEN
37 943 --                          gLongOffset.vh[vhs] := gPageOffset.vh[vhs]
37 944 --                      ELSE
37 945 --                          BEGIN
37 946 --                              gLongOffset.vh[vhs] := 0;
37 947 --                              theOrigin.vh[vhs] := theOrigin.vh[vhs] + gPageOffset.vh[vhs];
37 948 --                              aRect.topLeft.vh[vhs] := aRect.topLeft.vh[vhs] + gPageOffset.vh[vhs];
37 949 --                              aRect.botRight.vh[vhs] := aRect.botRight.vh[vhs] + gPageOffset.vh[vhs];
37 950 --                          END;
37 951 --                      END;
37 952 --              { $H+ }
37 953 --              SetOrigin(theOrigin.h, theOrigin.v);
37 954 --
37 955 --              { Clip the page to the intersection of the visible part of the view and the
37 956 --                printable area of the page. Note that in some cases (e.g., a WYSIWYG
37 957 --                word processor which showed the complete page margin) parts of
37 958 --                the interior might lie outside theInk. We must clip to theInk to
37 959 --                avoid slowing down PostScript printers. }
37 960 --              fView.ViewToQDRect(fViewedRect, rectToClip);
37 961 --              IF SectRect(rectToClip, aRect, rectToClip) THEN ;
37 962 --                  ClipRect(rectToClip);
37 963 --
37 964 -0 A  END;
37 965 --
37 966 --
37 967 --      { $S PrintRes }
37 968 --      {-----+
37 969 --      |   GetBreakCoord
37 970 --      +-----+}
37 971 -- A  PROCEDURE TStdPrintHandler.GetBreakCoord(vhs: VHSelect; whichBreak: INTEGER; VAR loc: VCoordinate);
37 972 --      VAR
37 973 --          automatic: BOOLEAN;
37 974 --          i:          INTEGER;
37 975 --          startBreak: INTEGER;
37 976 --          orthoVHS:   VHSelect;
37 977 0- A  BEGIN
37 978 --          orthoVHS := gOrthogonal[vhs];
37 979 --          IF fFixedSizePages[orthoVHS] THEN
37 980 --              loc := Min(fPrintExtent.topLeft.vh[orthoVHS] + fViewPerPage.vh[orthoVHS] * whichBreak,
37 981 --                          fPrintExtent.botRight.vh[orthoVHS])
37 982 --          ELSE IF whichBreak = fLastStrip.vh[vhs] THEN
37 983 --              loc := fLastBreak.vh[vhs]
37 984 --          ELSE
37 985 --              BEGIN
37 986 --                  IF whichBreak > fLastStrip.vh[vhs] THEN
37 987 --                      BEGIN
37 988 --                          startBreak := fLastStrip.vh[vhs] + 1;
37 989 --                          loc := fLastBreak.vh[vhs];

```

```

37 990 -2          END
37 991 --          ELSE
37 992 2-          BEGIN
37 993 --              startBreak := 1;
37 994 --              loc := fPrintExtent.topLeft.vh[gOrthogonal[vhs]];
37 995 -2          END;
37 996 --          FOR i := startBreak TO whichBreak DO
37 997 --              loc := fView.DoBreakFollowing(vhs, loc, automatic); { ??? error handling??? }
37 998 -1          END;
37 999 --
37 1000 --          fLastStrip.vh[vhs] := whichBreak;
37 1001 --          fLastBreak.vh[vhs] := loc;
37 1002 -0 A      END;
37 1003 --
37 1004 --
37 1005 --          {$$ PrintActual}
37 1006 --          {-----+
37 1007 --          |   GetDocName
37 1008 --          +-----}
37 1009 -- A      PROCEDURE TStdPrintHandler.GetDocName(VAR docName: Str255);
37 1010 --          VAR
37 1011 --              aWindow:   TWindow;
37 1012 0- A      BEGIN
37 1013 --              IF fDocument <> NIL THEN
37 1014 --                  docName := fDocument.fTitle
37 1015 --              ELSE
37 1016 --                  docName := '';
37 1017 --              IF docName = '' THEN
37 1018 1-          BEGIN
37 1019 --                  aWindow := fView.GetWindow;
37 1020 --                  IF aWindow <> NIL THEN
37 1021 --                      GetWTitle(aWindow.fWmgrWindow, docName);
37 1022 -1          END;
37 1023 --          {$IFC qDebug}
37 1024 --              IF docName = '' THEN
37 1025 --                  ProgramBreak('GetDocName can''t get a document or window name');
37 1026 --          {$ENDC}
37 1027 -0 A      END;
37 1028 --
37 1029 --
37 1030 --          {$IFC qDebug}
37 1031 --          {$$ PrintDebug}
37 1032 --          {-----+
37 1033 --          |   IdentifySoftware
37 1034 --          +-----}
37 1035 -- A      PROCEDURE TStdPrintHandler.IdentifySoftware;
37 1036 0- A      BEGIN
37 1037 --          WriteLn('UPrinting of ', COMPDATE, ' @ ', COMPTIME);
37 1038 -0 A      END;
37 1039 --          {$ENDC}
37 1040 --
37 1041 --
37 1042 --          {$$ PrintNonRes}
37 1043 --          {-----+
37 1044 --          |   InstallMargins
37 1045 --          +-----}
37 1046 -- A      PROCEDURE TStdPrintHandler.InstallMargins(newMargins: Rect; areMinimalMargins: BOOLEAN);
37 1047 0- A      BEGIN
37 1048 --          fMinimalMargins := areMinimalMargins;
37 1049 --
37 1050 --          IF fMinimalMargins THEN
37 1051 1-          BEGIN
37 1052 --              WITH fPageAreas DO
37 1053 2-          BEGIN
37 1054 --              {$H-}
37 1055 --              theMargins := theInk;
37 1056 --              SubPt(thePaper.topLeft, theMargins.topLeft);
37 1057 --              SubPt(thePaper.botRight, theMargins.botRight);
37 1058 --              {$H+}
37 1059 -2          END;
37 1060 --              WITH fPageAreas DO
37 1061 --                  theInterior := theInk;
37 1062 -1          END
37 1063 --          ELSE
37 1064 1-          BEGIN
37 1065 --              fPageAreas.theMargins := newMargins;
37 1066 --              WITH fPageAreas DO
37 1067 2-          BEGIN
37 1068 --                  theInterior := thePaper;
37 1069 --                  AddPt(theMargins.topLeft, theInterior.topLeft);
37 1070 --                  AddPt(theMargins.botRight, theInterior.botRight); {$H+}
37 1071 -2          END;
37 1072 -1          END;
37 1073 --
37 1074 -0 A      END;
37 1075 --
37 1076 --
37 1077 --          {$$ PrintNonRes}
37 1078 --          {-----+
37 1079 --          |   InvalPageFeedback
37 1080 --          +-----}
37 1081 -- A      PROCEDURE TStdPrintHandler.InvalPageFeedback;
37 1082 0- A      BEGIN
37 1083 --          IF ShowsOnScreen THEN
37 1084 --              fView.ForceRedraw;
37 1085 -0 A      END;
37 1086 --
37 1087 --
37 1088 --          {$$ PrintNonRes}

```



```

37 1188 1-      BEGIN
37 1189 --      IF THPrint(fHPrint)^^.PrJob.BJDocLoop = BSpoolLoop THEN
37 1190 --          IF NOT justSpool THEN
37 1191 --              PrintSpoolFile(fHPrint, err, proceed);
37 1192 -1      END;
37 1193 --      IF NOT proceed THEN
37 1194 --          IF err <> iprAbort THEN
37 1195 --              Failure(err, 0);
37 1196 -0 A  END;
37 1197 --
37 1198 --
37 1199 --      {$$ PrintRes}
37 1200 --      {-----+
37 1201 --      |   OpenPrintShop
37 1202 --      +-----}
37 1203 -- A  PROCEDURE TStdPrintHandler.OpenPrintShop;
37 1204 --
37 1205 --      VAR
37 1206 --          err:      OSErr;
37 1207 --
37 1208 0- A  BEGIN
37 1209 --          PrOpen;          { Open the print shop }
37 1210 --          err := PrError;   { Get code }
37 1211 --          IF err <> NoErr THEN
37 1212 1-      BEGIN
37 1213 --          IF (err = fnfErr) OR (err = resFNotFound) THEN
37 1214 --              err := errNoPrintDrvr;
37 1215 --              Failure(err, 0);
37 1216 -1      END;
37 1217 -0 A  END;
37 1218 --
37 1219 --
37 1220 --      {$$ PrintRes}
37 1221 --      {-----+
37 1222 --      |   PageToStrip
37 1223 --      +-----}
37 1224 -- A  FUNCTION TStdPrintHandler.PageToStrip (pageNumber: INTEGER): Point;
37 1225 --
37 1226 --      VAR
37 1227 --          normalizedPageNum:  INTEGER;
37 1228 --          ortho:              VHSelect;
37 1229 --          strip:              Point;
37 1230 0- A  BEGIN
37 1231 --          normalizedPageNum := pageNumber - fStartPage + 1;
37 1232 --
37 1233 --          ortho := gOrthogonal[fPageDirection];
37 1234 --
37 1235 --          strip.vh[ortho] := ((normalizedPageNum - 1) DIV fPageStrips.vh[ortho]);
37 1236 --          strip.vh[fPageDirection] := normalizedPageNum -
37 1237 --              ((strip.vh[ortho]) * fPageStrips.vh[ortho]) - 1;
37 1238 --
37 1239 --          PageToStrip := strip;
37 1240 -0 A  END;
37 1241 --
37 1242 --
37 1243 --      {$$ PrintRes}
37 1244 --      {-----+
37 1245 --      |   PointToPageStrip
37 1246 --      +-----}
37 1247 -- A  FUNCTION TStdPrintHandler.PointToPageStrip(pointInView: VPoint): Point;
37 1248 --      { !!! Need to test this! }
37 1249 --      VAR
37 1250 --          currCoord:  INTEGER;
37 1251 --          automatic:  BOOLEAN;
37 1252 --          aStrip:     INTEGER;
37 1253 --          vhs:        VHSelect;
37 1254 --          pageStrip:  Point;
37 1255 0- A  BEGIN
37 1256 --          FOR vhs := v TO h DO
37 1257 --              IF fFixedSizePages[vhs] THEN
37 1258 --                  pageStrip.vh[vhs] := (PointInView.vh[vhs] DIV fViewPerPage.vh[vhs]) + 1
37 1259 --              ELSE
37 1260 1-      BEGIN
37 1261 --                  pageStrip.vh[vhs] := 1;
37 1262 --                  currCoord := fPrintExtent.topLeft.vh[vhs]; { ??? or orthogonal? }
37 1263 --                  WHILE pointInView.vh[vhs] > currCoord DO { ??? or orthogonal? }
37 1264 2-      BEGIN
37 1265 --                  currCoord := fView.DoBreakFollowing(vhs, currCoord, automatic);
37 1266 --                  pageStrip.vh[vhs] := aStrip + 1;
37 1267 -2      END;
37 1268 -1      END;
37 1269 --                  PointToPageStrip := pageStrip;
37 1270 -0 A  END;
37 1271 --
37 1272 --
37 1273 --      {$$ PrintActual}
37 1274 --      {-----+
37 1275 --      |   PoseJobDialog
37 1276 --      +-----}
37 1277 -- A  FUNCTION TStdPrintHandler.PoseJobDialog: BOOLEAN;
37 1278 --      VAR
37 1279 --          b:      BOOLEAN;
37 1280 --          proceed: BOOLEAN;
37 1281 --          temp:   INTEGER;
37 1282 --          err:    OSErr;
37 1283 --
37 1284 --      {-----+
37 1285 --      |   CallJobDialog
37 1286 --      +-----}

```

```

37 1287 -- B      PROCEDURE CallJobDialog;
37 1288 0- B      BEGIN
37 1289 --          proceed := PrJobDialog(THPrint(fHPrint));
37 1290 -0 B      END;
37 1291 --
37 1292 0- A      BEGIN
37 1293 --          proceed := TRUE;
37 1294 --          (* PrepareForDialog; *)
37 1295 --          DoInMacPrint(CallJobDialog);
37 1296 --          ChkPrintErr(err, proceed, b);
37 1297 --
37 1298 --          WITH THPrint(fHPrint)^^.prJob DO
37 1299 --              IF iFstPage > iLstPage THEN { Rectify the range as a public service if needed }
37 1300 1-                  BEGIN
37 1301 --                      temp := iLstPage;
37 1302 --                      iLstPage := iFstPage;
37 1303 --                      iFstPage := temp;
37 1304 -1                  END;
37 1305 --
37 1306 --          gApplication.UpdateAllWindows;
37 1307 --          PoseJobDialog := proceed;
37 1308 -0 A      END;
37 1309 --
37 1310 --
37 1311 --          {$S PrintActual}      { Needs to be here because it may be called from Print }
37 1312 --          {-----+
37 1313 --          |   PosePageSetupDialog   |
37 1314 --          +-----}
37 1315 -- A      FUNCTION TStdPrintHandler.PosePageSetupDialog(
37 1316 --          VAR proceed: BOOLEAN; isUndoable: BOOLEAN): TCommand;
37 1317 --      VAR
37 1318 --          react: BOOLEAN;
37 1319 --          aPrintStyleChangeCommand: TPrintStyleChangeCommand;
37 1320 --
37 1321 --          {-----+
37 1322 --          |   CallStyleDialog   |
37 1323 --          +-----}
37 1324 -- B      PROCEDURE CallStyleDialog;
37 1325 0- B      BEGIN
37 1326 --          react := PrStdDialog(THPrint(fHPrint));
37 1327 -0 B      END;
37 1328 --
37 1329 0- A      BEGIN
37 1330 --          PosePageSetupDialog := gNoChanges;
37 1331 --          react := FALSE; { in case MacPrint code not accessible }
37 1332 --          IF NOT isUndoable THEN
37 1333 1-              BEGIN
37 1334 --                  (* PrepareForDialog; *)
37 1335 --                  DoInMacPrint(CallStyleDialog);
37 1336 --                  IF react THEN
37 1337 --                      CheckPrinter;
37 1338 -1              END
37 1339 --          ELSE
37 1340 1-              BEGIN
37 1341 --                  { Must setup command before putting up page setup dialog because the
37 1342 --                    command records the current print record to make it undoable. }
37 1343 --                  New(aPrintStyleChangeCommand);
37 1344 --                  FailNIL(aPrintStyleChangeCommand);
37 1345 --                  aPrintStyleChangeCommand.IPrintStyleChangeCommand(SELF);
37 1346 --
37 1347 --                  (* PrepareForDialog; *)
37 1348 --                  DoInMacPrint(CallStyleDialog); { Put up the Page Setup Dialog }
37 1349 --
37 1350 --                  IF react THEN { User specified a change }
37 1351 2-                      BEGIN
37 1352 --                          BlockMove(fHPrint^, aPrintStyleChangeCommand.fNewHPrint^,
37 1353 --                              SIZEOF(TPrint));
37 1354 --                          PosePageSetupDialog := aPrintStyleChangeCommand;
37 1355 -2                      END
37 1356 --                  ELSE { User did not specify a change }
37 1357 --                      aPrintStyleChangeCommand.Free;
37 1358 --
37 1359 --                  END;
37 1360 -1          proceed := react;
37 1361 --      END;
37 1362 -0 A      END;
37 1363 --
37 1364 --
37 1365 --          {$S PrintActual}
37 1366 --          {-----+
37 1367 --          |   PosePrintDialog   |
37 1368 --          +-----}
37 1369 -- A      PROCEDURE TStdPrintHandler.PosePrintDialog;
37 1370 --      VAR
37 1371 --          dlgNumber:      INTEGER;
37 1372 --          docName:        Str255;
37 1373 --          itemNo:         INTEGER;
37 1374 --          theItem:        Handle;
37 1375 --          itemType:       INTEGER;
37 1376 --          box:            Rect;
37 1377 --          itemText:       Str255;
37 1378 --          preDocName,
37 1379 --          constTitle:     INTEGER;
37 1380 --
37 1381 0- A      BEGIN
37 1382 --          IF gFinderPrinting THEN
37 1383 1-              BEGIN
37 1384 --                  dlgNumber := phFinderPrintDialog;
37 1385 --                  itemNo := 3;

```





```

37 1386 -1      END
37 1387 --      ELSE
37 1388 1-      BEGIN
37 1389 --          dlgNumber := phSpoolPrintDialog;
37 1390 --          itemNo := 2;
37 1391 -1      END;
37 1392 --      gPrintDialog := GetNewDialog(dlgNumber, NIL, POINTER(-1));
37 1393 --
37 1394 --      { Substitute the document name for '<<<>>' in 'Document "<<<>>" is being printed'.
37 1395 --      ParamText would be a lot simpler, but the Print Mgr. also uses ParamText, and
37 1396 --      the substitution doesn't happen until draw time.)
37 1397 --      GetDocName(docName);
37 1398 --      GetDitem(gPrintDialog, ItemNo, itemType, theItem, box);
37 1399 --      IF theItem <> NIL THEN
37 1400 1-          BEGIN
37 1401 --              GetIText(theItem, itemText);
37 1402 --              IF ParseTitleTemplate(itemText, preDocName, constTitle) &
37 1403 --              SubstituteInTitle(itemText, docName, preDocName, constTitle) THEN
37 1404 --                  SetIText(theItem, itemText);
37 1405 -1          END;
37 1406 --
37 1407 --      {IFC qDebug}
37 1408 --      IF gPrintDialog = NIL THEN
37 1409 1-          BEGIN
37 1410 --              ProgramBreak('TStdPrintHandler.PosePrintDialog: Could not load print dialog. ');
37 1411 --              EXIT(PosePrintDialog);
37 1412 -1          END;
37 1413 --      {SENDC}
37 1414 --
37 1415 --      THPrint(fhPrint)^^.prJob.pIdleProc := @CheckButton;
37 1416 --      SetWTitle(gPrintDialog, docName); { In case Print Mgr. needs it. }
37 1417 --      DrawDialog(gPrintDialog);
37 1418 -0 A  END;
37 1419 --
37 1420 --
37 1421 --      {$S PrintRes}
37 1422 --      { ???Was in PrintActual. Doesn't really belong in this segment, but
37 1423 --      can get unloaded during Finder printing otherwise. }
37 1424 --      {-----+
37 1425 --      | Print |
37 1426 --      +-----}
37 1427 -- A  FUNCTION TStdPrintHandler.Print(itsCmdNumber: CmdNumber; VAR proceed: BOOLEAN): TCommand; OVERRIDE;
37 1428 --      VAR
37 1429 --          fi:          FailInfo;
37 1430 --
37 1431 --      {-----+
37 1432 --      | DoPrint |
37 1433 --      +-----}
37 1434 -- B  PROCEDURE DoPrint;
37 1435 --      VAR
37 1436 --          firstPage:    INTEGER;
37 1437 --          lastPage:     INTEGER;
37 1438 --          err:          OSerr;
37 1439 --          spoolFileName: Str255;
37 1440 --          spoolVrefNum: INTEGER;
37 1441 --          aPrJob:       TPrJob;
37 1442 --          totalPages:   INTEGER;
37 1443 --          lastPrinted:  INTEGER;
37 1444 --          lastAttempted: INTEGER;
37 1445 --          pagesPerSubjob: INTEGER;
37 1446 --          firstSubjobPage: INTEGER;
37 1447 --          proceed:      BOOLEAN;
37 1448 --          ranOutOfSpace: BOOLEAN;
37 1449 --          spoolMethod:  BOOLEAN;
37 1450 --          justSpool:    BOOLEAN;
37 1451 --          dontCare:     BOOLEAN;
37 1452 --          pageStrips:   Point;
37 1453 --          fi:           FailInfo;
37 1454 --
37 1455 --      {-----+
37 1456 --      | Hd1SubjobFailure |
37 1457 --      +-----}
37 1458 -- C  PROCEDURE Hd1SubjobFailure(error: OSerr; message: LONGINT);
37 1459 0- C  BEGIN
37 1460 --      BanishPrintDialog;
37 1461 -0 C  END;
37 1462 --
37 1463 0- B  BEGIN
37 1464 --      justSpool := (itsCmdNumber = cPrintToFile);
37 1465 --      proceed := TRUE;
37 1466 --      ranOutOfSpace := FALSE;
37 1467 --      aPrJob := THPrint(fhPrint)^^.prJob;
37 1468 --      fView.DoCalcPageStrips(pageStrips);
37 1469 --      fPageStrips := pageStrips;
37 1470 --
37 1471 --      firstPage := Max(aPrJob.iFstPage, fStartPage);
37 1472 --      lastPage := Min(aPrJob.iLstPage, MaxPageNumber);
37 1473 --
37 1474 --      IF lastPage < firstPage THEN
37 1475 --          err := Alert(phNoPages, NIL)
37 1476 --      ELSE
37 1477 1-          BEGIN
37 1478 --              totalPages := lastPage - firstPage + 1;
37 1479 --              spoolMethod := (aPrJob.BJDocLoop = BSpoolLoop);
37 1480 --              IF spoolMethod THEN
37 1481 2-                  BEGIN
37 1482 --                      ChooseSpoolFile(spoolFileName, spoolVrefNum, pagesPerSubjob);
37 1483 --                      IF NOT justSpool THEN { if justSpool is true, then the spool filename & vrefNum will
37 1484 --                      already have been stuffed into the prJob record before

```



```

37 1485 --                                this method is called )
37 1486 3-                                BEGIN
37 1487 --                                IF spoolFileName <> '' THEN
37 1488 --                                    WITH THPrint(fhPrint)^^.prJob DO
37 1489 4-                                        BEGIN
37 1490 --                                            pFileName := @spoolFileName;
37 1491 --                                            iFileVol := spoolVRefNum;
37 1492 4-                                            END;
37 1493 3-                                        END;
37 1494 2-                                    END
37 1495 --                                ELSE
37 1496 --                                    pagesPerSubjob := MAXINT;
37 1497 --
37 1498 --                                    lastPrinted := firstPage - 1;
37 1499 --
37 1500 --                                    pagesPerSubjob := Min(pagesPerSubjob, totalPages);
37 1501 --                                    PosePrintDialog;
37 1502 --
37 1503 --                                    CatchFailures(fi, HdlSubjobFailure);
37 1504 2-                                REPEAT
37 1505 --                                    firstSubjobPage := lastPrinted + 1;
37 1506 --                                    Print := OneSubJob(firstSubjobPage,
37 1507 --                                        firstSubjobPage + pagesPerSubjob - 1,
37 1508 --                                        justSpool, (pagesPerSubjob < totalPages), ranOutOfSpace,
37 1509 --                                        lastAttempted, proceed);
37 1510 --                                    IF proceed THEN
37 1511 --                                        lastPrinted := lastAttempted;
37 1512 --                                    IF ranOutOfSpace THEN
37 1513 3-                                        BEGIN
37 1514 --                                            pagesPerSubjob := lastAttempted - 1 - firstSubjobPage;
37 1515 --                                            proceed := TRUE;
37 1516 3-                                        END;
37 1517 2-                                UNTIL
37 1518 --                                    (lastPrinted = lastPage) OR (pagesPerSubjob < 1) OR NOT proceed;
37 1519 --
37 1520 --                                IF pagesPerSubjob < 1 THEN
37 1521 --                                    Failure(errSpooling, 0);
37 1522 --                                    Success(fi);
37 1523 --                                    BanishPrintDialog; ( having removed, put back here, unsure if exactly right )
37 1524 1-                                END; ( if there are pages within requested range )
37 1525 --
37 1526 0- B                                END;
37 1527 --
37 1528 --                                {-----+
37 1529 --                                |   HdlPrintFailure   |
37 1530 --                                +-----+
37 1531 0- B                                PROCEDURE HdlPrintFailure(error: OSErr; message: LONGINT);
37 1532 0- B                                BEGIN
37 1533 --                                    { Certain Print Manager errors should not result in any alert,
37 1534 --                                        since the Print Manager will have already put one up. }
37 1535 --                                    IF (error >= -8160) AND (error <= -8150) THEN
37 1536 --                                        Failure(0, msgPrintFailed);
37 1537 --                                    IF message = 0 THEN
37 1538 --                                        GetDocName(gErrorParm3);
37 1539 --                                        FailNewMessage(error, message, msgPrintFailed);
37 1540 0- B                                END;
37 1541 --
37 1542 0- A                                BEGIN { TStdPrintHandler.Print }
37 1543 --                                    Print := qNoChanges;
37 1544 --                                    gCancelAllPrinting := FALSE;
37 1545 --
37 1546 --                                    SetPrintExtent;      { Make sure we've got the right area. }
37 1547 --
37 1548 --                                    CatchFailures(fi, HdlPrintFailure);
37 1549 --                                    DoInMacPrint(DoPrint);
37 1550 --                                    Success(fi);
37 1551 --                                    proceed := NOT gCancelAllPrinting;
37 1552 0- A                                END;
37 1553 --
37 1554 --
37 1555 --                                {$S PrintNonRes}
37 1556 --                                {-----+
37 1557 --                                |   PrinterChanged   |
37 1558 --                                +-----+
37 1559 0- A                                PROCEDURE TStdPrintHandler.PrinterChanged; OVERRIDE;
37 1560 0- A                                BEGIN
37 1561 --                                    fView.DoPagination;
37 1562 0- A                                END;
37 1563 --
37 1564 --
37 1565 --                                {$S PrintImage}
37 1566 --                                {-----+
37 1567 --                                |   PrintPage   |
37 1568 --                                +-----+
37 1569 0- A                                PROCEDURE TStdPrintHandler.PrintPage(aPageNumber: INTEGER); { Print a single page }
37 1570 --                                VAR
37 1571 --                                ($IFC qDebug)
37 1572 --                                aPort: GrafPtr;
37 1573 --                                ($ENDC)
37 1574 --                                fi:      FailInfo;
37 1575 --
37 1576 --                                {-----+
37 1577 --                                |   CheckPrintFailure   |
37 1578 --                                +-----+
37 1579 0- B                                PROCEDURE CheckPrintFailure;
37 1580 0- B                                BEGIN
37 1581 --                                    FailOSErr(PrError);
37 1582 0- B                                END;
37 1583 --

```

```

37 1584 --      {-----+
37 1585 --      | HdlPrintFailure |
37 1586 --      +-----+
37 1587 -- B      PROCEDURE HdlPrintFailure(error: OSErr; message: LONGINT);
37 1588 -- B      BEGIN
37 1589 --      PrClosePage(gPPrPort);
37 1590 -- B      END;
37 1591 --
37 1592 -- A      BEGIN
37 1593 --      SetPage(aPageNumber); { gets gPage set up correctly for coordinate transformations }
37 1594 --      CatchFailures(fi, HdlPrintFailure);
37 1595 --      PrOpenPage(gPPrPort, NIL);
37 1596 --      CheckPrintFailure;
37 1597 --      FocusOnInterior;
37 1598 --      DrawPageInterior;
37 1599 --      {SIFC qDebug}
37 1600 --      GetPort(aPort);
37 1601 --      IF aPort <> GrafPtr(gPPrPort) THEN
37 1602 --          ProgramBreak('The view's DrawPageInterior method changed the grafPort');
37 1603 --      {SENDC}
37 1604 --      CheckPrintFailure;
37 1605 --      FocusOnBorder;
37 1606 --      AdornPage;
37 1607 --      CheckPrintFailure;
37 1608 --      Success(fi);
37 1609 --      PrClosePage(gPPrPort);
37 1610 -- A      END;
37 1611 --
37 1612 --
37 1613 --      {$S PrintNonRes}
37 1614 --      {-----+
37 1615 --      | RedoPageBreaks |
37 1616 --      +-----+
37 1617 -- A      PROCEDURE TStdPrintHandler.RedoPageBreaks; OVERRIDE;
37 1618 --      { Called from fView.DoPagination. }
37 1619 --
37 1620 --      VAR
37 1621 --          worryAboutBreaks:   BOOLEAN;
37 1622 --          oldViewPerPage,
37 1623 --          viewPerPage:        VPoint;
37 1624 --          pageStrips:         Point;
37 1625 --          newInterior,
37 1626 --          oldInterior:        Rect;
37 1627 -- A      BEGIN
37 1628 --          worryAboutBreaks := (fView.GetGrafPort <> NIL) AND gInitialized AND (fShowBreaks OR gDebugPrinting);
37 1629 --
37 1630 --          IF worryAboutBreaks THEN
37 1631 --              InvalPageFeedback; { invalidate old page breaks, if relevant }
37 1632 --
37 1633 --          SetPrintExtent;
37 1634 --          oldInterior := fPageAreas.theInterior;
37 1635 --          oldViewPerPage := fViewPerPage;
37 1636 --          SetMargins;
37 1637 --
37 1638 --          { computes view per page from papersize, printer resolution, margins desired, view resolution,
37 1639 --            and, if desired, other factors such as printable rectangle of page and font metrics in the
37 1640 --            printer space }
37 1641 --          fView.DoCalcViewPerPage(viewPerPage);
37 1642 --          fViewPerPage := viewPerPage;
37 1643 --          SetPageInterior(kUsualPages);
37 1644 --          newInterior := fPageAreas.theInterior;
37 1645 --
37 1646 --          {SIFC qDebug}
37 1647 --          IF gDebugPrinting THEN
37 1648 --              IF NOT EqualRect(oldInterior, newInterior) THEN
37 1649 --                  ProgramBreak('Setting new interior');
37 1650 --          {SENDC}
37 1651 --
37 1652 --          IF NOT EqualRect(oldInterior, newInterior) THEN
37 1653 --              fView.PageInteriorChanged(newInterior);
37 1654 --
37 1655 --          {SIFC qDebug}
37 1656 --          IF gDebugPrinting THEN
37 1657 --              IF NOT EqualVpt(oldViewPerPage, viewPerPage) THEN
37 1658 --                  ProgramBreak('Setting new view per page');
37 1659 --          {SENDC}
37 1660 --
37 1661 --          IF NOT EqualRect(oldInterior, newInterior) |
37 1662 --              NOT EqualVpt(oldViewPerPage, viewPerPage) THEN
37 1663 --              fView.AdjustSize;
37 1664 --
37 1665 --          fView.DoCalcPageStrips(pageStrips);
37 1666 --          fPageStrips := pageStrips;
37 1667 --
37 1668 --          IF worryAboutBreaks THEN
37 1669 --              InvalPageFeedback; { force redraw of new page breaks, if relevant }
37 1670 --
37 1671 -- A      END;
37 1672 --
37 1673 --
37 1674 --      {$S PrintNonRes}
37 1675 --      {-----+
37 1676 --      | Reset |
37 1677 --      +-----+
37 1678 -- A      PROCEDURE TStdPrintHandler.Reset;
37 1679 --      { subclass may stuff some special values by redefining this }
37 1680 --
37 1681 --      LABEL
37 1682 --          1000;

```

```

37 1683 --
37 1684 -- TYPE
37 1685 --     TXWord = PACKED RECORD
37 1686 --     CASE INTEGER OF
37 1687 --         0: (c1, c0: CHAR);
37 1688 --         1: (b1, b0: SignedByte);
37 1689 --         2: (f15, f14, f13, f12, f11, f10, f9, f8, f7, f6, f5, f4, f3, f2, f1, f0: BOOLEAN);
37 1690 --         3: (i0: INTEGER);
37 1691 --     END;
37 1692 --
37 1693 -- VAR
37 1694 --     anHPrint:    THPrint;
37 1695 --     didChange:    BOOLEAN;
37 1696 --     initd:        BOOLEAN;
37 1697 --     fi:           FailInfo;
37 1698 --
37 1699 --     {-----+
37 1700 --     | CallPrintDefault |
37 1701 --     +-----+}
37 1702 -- B PROCEDURE CallPrintDefault;
37 1703 0- B BEGIN
37 1704 --     PrintDefault(anHPrint);
37 1705 -0 B END;
37 1706 --
37 1707 --     {-----+
37 1708 --     | HdlDefaultError |
37 1709 --     +-----+}
37 1710 -- B PROCEDURE HdlDefaultError(error: OSErr; message: LONGINT);
37 1711 0- B BEGIN
37 1712 --     GOTO 1000;
37 1713 -0 B END;
37 1714 --
37 1715 0- A BEGIN
37 1716 --     anHPrint := THPrint(fhPrint);
37 1717 --     initd := FALSE;
37 1718 --     IF anHPrint <> NIL THEN
37 1719 1- BEGIN
37 1720 --         IF gCouldPrint THEN
37 1721 2- BEGIN
37 1722 --             CatchFailures(fi, HdlDefaultError);
37 1723 --             DoInMacPrint(CallPrintDefault);
37 1724 --             IF fSquareDots THEN
37 1725 --                 WITH TXWord(THPrint(anHPrint)^^.prStl.wDev) DO
37 1726 --                     IF b1 = 1 THEN { 1 = bDevCtch => ImageWriter }
37 1727 3- BEGIN
37 1728 --                         { Set the square-pixel flag in the print record, then
37 1729 --                         ensure we didn't corrupt it. }
37 1730 --                         f2 := TRUE;
37 1731 --                         ValidatePrintRecord(didChange);
37 1732 -3 END;
37 1733 --                         initd := TRUE;
37 1734 --                         Success(fi);
37 1735 -2 END;
37 1736 -- 1000:
37 1737 --     IF NOT (gCouldPrint AND initd) THEN
37 1738 --         { MacPrint code unavailable, but we want some plausible things
37 1739 --         anyway - set up for Portrait Tall Adj. }
37 1740 2- BEGIN
37 1741 --         { ??? Replace the following expensive stuffing code with StuffHex
37 1742 --         calls or a GetResource or some such, later }
37 1743 --         WITH anHPrint^^ DO {SH-}
37 1744 3- BEGIN
37 1745 --             iPrVersion := 0; { something invalid }
37 1746 --
37 1747 --             WITH prInfo DO
37 1748 4- BEGIN
37 1749 --                 iHRes := 72;
37 1750 --                 iVRes := 72;
37 1751 --                 SetRect(rPage, 0, 0, 576, 752); { must have its top left @ (0,0) }
37 1752 -4 END;
37 1753 --
37 1754 --                 SetRect(rPaper, -18, -36, 594, 756);
37 1755 --
37 1756 --                 WITH prStl DO
37 1757 4- BEGIN
37 1758 --                     iPageV := 1320; { 11 inches in 120th of an inch }
37 1759 --                     iPageH := 1020; { 8.5 inches in 120th of an inch }
37 1760 -4 END;
37 1761 -3 END; {SH+}
37 1762 -2 END; { IF NOT (gCouldPrint AND initd) }
37 1763 --
37 1764 -1 END; { IF anHPrint <> NIL }
37 1765 -0 A END;
37 1766 --
37 1767 --
37 1768 --     {$$ PrintNonRes}
37 1769 --     {-----+
37 1770 --     | SetPrintExtent |
37 1771 --     +-----+}
37 1772 -- A PROCEDURE TStdPrintHandler.SetPrintExtent;
37 1773 -- VAR
37 1774 --     printExtent: VRect;
37 1775 0- A BEGIN
37 1776 --     fView.GetPrintExtent(printExtent);
37 1777 --     { Make sure the bottom or right is not smaller than the top or left.
37 1778 --     This can happen when views are being created and before the window
37 1779 --     has been resized to its actual size. }
37 1780 --     printExtent.bottom := Max(printExtent.bottom, printExtent.top);
37 1781 --     printExtent.right := Max(printExtent.right, printExtent.left);

```



```

37 1782 --      fPrintExtent := printExtent;
37 1783 -0 A  END;
37 1784 --
37 1785 --
37 1786 --      {$S PrintOpen}
37 1787 --      {-----+
37 1788 --      | SetDefaultPrintInfo |
37 1789 --      +-----}
37 1790 -- A  PROCEDURE TStdPrintHandler.SetDefaultPrintInfo;
37 1791 --      VAR
37 1792 --          didChange:    BOOLEAN;
37 1793 --          wantValidate:  BOOLEAN;
37 1794 --          anHPrint:      THPrint;
37 1795 --          haveHPrint:    BOOLEAN;
37 1796 0- A  BEGIN
37 1797 --          anHPrint := THPrint(fHPrint);
37 1798 --          IF anHPrint <> NIL THEN
37 1799 --              DisposHandle(Handle(anHPrint));
37 1800 --          wantValidate := FALSE;
37 1801 --
37 1802 --          haveHPrint := FALSE;
37 1803 --          IF fView <> NIL THEN
37 1804 1-              BEGIN
37 1805 --                  IF fDocument <> NIL THEN
37 1806 --                      IF fDocument.fSharePrintInfo AND (fDocument.fPrintInfo <> NIL) THEN
37 1807 2-                          BEGIN
37 1808 --                              fHPrint := fDocument.fPrintInfo;
37 1809 --                              haveHPrint := TRUE;
37 1810 -2                          END;
37 1811 -1                      END;
37 1812 --
37 1813 --          IF haveHPrint THEN
37 1814 --              wantValidate := TRUE
37 1815 --          ELSE
37 1816 1-              BEGIN
37 1817 --                  fHPrint := NewPermHandle(SIZEOF(TPrint));
37 1818 --                  FailNIL(fHPrint);
37 1819 --                  Reset;
37 1820 -1                  END;
37 1821 --
37 1822 --          IF wantValidate THEN
37 1823 --              ValidatePrintRecord(didChange);
37 1824 -0 A  END;
37 1825 --
37 1826 --
37 1827 --      {$S PrintNonRes}
37 1828 --      {-----+
37 1829 --      | SetMargins |
37 1830 --      +-----}
37 1831 -- A  PROCEDURE TStdPrintHandler.SetMargins;
37 1832 --      VAR
37 1833 --          someMargins:  Rect;
37 1834 0- A  BEGIN
37 1835 --          someMargins := fPageAreas.theMargins;          { Prevent heap scramble }
37 1836 --          InstallMargins(someMargins, fMinimalMargins);
37 1837 -0 A  END;
37 1838 --
37 1839 --
37 1840 --      {$S PrintImage}
37 1841 --      {-----+
37 1842 --      | SetPage |
37 1843 --      +-----}
37 1844 -- A  PROCEDURE TStdPrintHandler.SetPage(aPageNumber: INTEGER);
37 1845 --      VAR
37 1846 --          viewedRect: VRect;
37 1847 --          vhs:        VHSelect;
37 1848 --          strip:       Point;
37 1849 0- A  BEGIN
37 1850 --          fFocusedPage := aPageNumber;
37 1851 --
37 1852 --          strip := PageToStrip(aPageNumber);
37 1853 --          FOR vhs := v TO h DO
37 1854 1-              BEGIN
37 1855 --                  GetBreakCoord(gOrthogonal[vhs], strip.vh[vhs], viewedRect.topLeft.vh[vhs]);
37 1856 --                  GetBreakCoord(gOrthogonal[vhs], strip.vh[vhs] + 1, viewedRect.botRight.vh[vhs]);
37 1857 -1                  END;
37 1858 --
37 1859 --          SetPageInterior(aPageNumber);
37 1860 --          fView.DoSetPageOffset(viewedRect.topLeft);
37 1861 --          fViewedRect := viewedRect;
37 1862 --
37 1863 --      {$IFC qDebug}
37 1864 --          IF gDebugPrinting THEN
37 1865 1-              BEGIN
37 1866 --                  Write('pg #: ', aPageNumber:1);
37 1867 --                  Write(' coord = ');
37 1868 --                  WriteVpt(viewedRect.topLeft);
37 1869 --                  Write('Page Interior: ');
37 1870 --                  WriteRect(fPageAreas.theInterior);{ ???HS }
37 1871 --                  WriteLn;
37 1872 -1                  END;
37 1873 --      {$ENDC}
37 1874 -0 A  END;
37 1875 --
37 1876 --
37 1877 --      {$S PrintNonRes}
37 1878 --      {-----+
37 1879 --      | SetPageInterior |
37 1880 --      +-----}

```





```

37 1881 -- A  PROCEDURE TStdPrintHandler.SetPageInterior(pageNumber: INTEGER); OVERRIDE;
37 1882 --  VAR
37 1883 --      vhs:      VHSelect;
37 1884 --      pegPoint:  Point;
37 1885 0- A  BEGIN
37 1886 --      WITH fPageAreas, theInterior DO
37 1887 1-      BEGIN
37 1888 --          FOR vhs := v TO h DO
37 1889 2-      BEGIN
37 1890 --          topLeft.vh[vhs] := thePaper.topLeft.vh[vhs] + theMargins.topLeft.vh[vhs];
37 1891 --          botRight.vh[vhs] := topLeft.vh[vhs] + fViewPerPage.vh[vhs];
37 1892 --      END;
37 1893 -1      END;
37 1894 --
37 1895 --      LocatePageInterior(pageNumber, pegPoint);
37 1896 --      WITH fPageAreas.theInterior DO
37 1897 1-      BEGIN
37 1898 --          topLeft := pegPoint;
37 1899 --          FOR vhs := v TO h DO
37 1900 --              botRight.vh[vhs] := topLeft.vh[vhs] + fViewPerPage.vh[vhs];
37 1901 -1      END;
37 1902 -0 A  END;
37 1903 --
37 1904 --
37 1905 --      {$S PrintImage}
37 1906 --      {-----+
37 1907 --      |  SetPageOffset      |
37 1908 --      +-----+}
37 1909 -- A  PROCEDURE TStdPrintHandler.SetPageOffset(coord: VPoint); OVERRIDE;
37 1910 --  VAR
37 1911 --      vhs:      VHSelect;
37 1912 0- A  BEGIN
37 1913 --      FOR vhs := v TO h DO
37 1914 --          gPageOffset.vh[vhs] := coord.vh[vhs] -
37 1915 --              fPageAreas.theInterior.topLeft.vh[vhs];
37 1916 -0 A  END;
37 1917 --
37 1918 --
37 1919 --      {$S PrintFinder}
37 1920 --      {-----+
37 1921 --      |  SetupForFinder      |
37 1922 --      +-----+}
37 1923 -- A  FUNCTION TStdPrintHandler.SetupForFinder: BOOLEAN; OVERRIDE;
37 1924 --  VAR
37 1925 --      proceed:      BOOLEAN;
37 1926 --      didChange:     BOOLEAN;
37 1927 0- A  BEGIN
37 1928 --      proceed := TRUE;
37 1929 --      CheckPrinter;      { Uncertain if necessary??? }
37 1930 --
37 1931 --      IF fFinderSetup THEN
37 1932 --          IF PosePageSetupDialog(proceed, FALSE) <> gNoChanges THEN
37 1933 --              {$IFC qDebug}
37 1934 --                  ProgramBreak('PosePageSetupDialog returned real command.')
37 1935 --              {$ENDC}
37 1936 --
37 1937 --      IF proceed THEN
37 1938 1-      BEGIN
37 1939 --          ShowDocBeingPrinted(TRUE);
37 1940 --
37 1941 --          IF fFinderJobDialog OR (gFinderHPrint = NIL) THEN
37 1942 2-      BEGIN
37 1943 --          proceed := PoseJobDialog;
37 1944 --          { Merge into gPrintHandler, in case next document needs it }
37 1945 --          IF gFinderHPrint = NIL THEN
37 1946 3-      BEGIN
37 1947 --              gFinderHPrint := NewPermHandle(SIZEOF(TPrint));
37 1948 --              FailNIL(gFinderHPrint);
37 1949 -3      END;
37 1950 --              BlockMove(fHPrint^, gFinderHPrint^, SIZEOF(TPrint));
37 1951 -2      END
37 1952 --          ELSE
37 1953 2-      BEGIN
37 1954 --              PrJobMerge(THPrint(gFinderHPrint), THPrint(fHPrint));
37 1955 --              ValidatePrintRecord(didChange);
37 1956 -2      END;
37 1957 --
37 1958 --          ShowDocBeingPrinted(FALSE);
37 1959 -1      END;
37 1960 --          SetupForFinder := proceed;
37 1961 -0 A  END;
37 1962 --
37 1963 --
37 1964 --      {$S PrintRes}
37 1965 --      {-----+
37 1966 --      |  SetupPrintOne      |
37 1967 --      +-----+}
37 1968 -- A  FUNCTION TStdPrintHandler.SetupPrintOne: BOOLEAN;
37 1969 --  VAR
37 1970 --      didChange:      BOOLEAN;
37 1971 0- A  BEGIN
37 1972 --      { Call PrValidate to 1) make sure the record is OK, and
37 1973 --      2) Get the LaserWriter driver to get the name of the
37 1974 --      front window. }
37 1975 --      ValidatePrintRecord(didChange);
37 1976 --
37 1977 --      PrSetError(noErr);
37 1978 --      WITH THPrint(fHPrint)^.prJob DO
37 1979 1-      BEGIN

```

```

37 1980 --      ifstPage := 0;
37 1981 --      ilstPage := 9999;      { This is what the Print Mgr. uses }
37 1982 -1      END;
37 1983 --      SetupPrintOne := TRUE;      { Should always be able to continue from here }
37 1984 -0 A  END;
37 1985 --
37 1986 --
37 1987 --      {$S PrintFinder}
37 1988 --      {-----+
37 1989 --      |   ShowDocBeingPrinted   |
37 1990 --      +-----}
37 1991 -- A  PROCEDURE TStdPrintHandler.ShowDocBeingPrinted (entering: BOOLEAN);
37 1992 --      VAR
37 1993 --          aTitle:      Str255;
37 1994 0- A  BEGIN
37 1995 --          IF entering THEN
37 1996 1-      BEGIN
37 1997 --          gPrintDialog := GetNewDialog(phWhichDoc, NIL, POINTER(-1));
37 1998 --          IF gPrintDialog <> NIL THEN
37 1999 2-      BEGIN
37 2000 --              GetDocName(aTitle);
37 2001 --              SetWTitle(gPrintDialog, aTitle);
37 2002 --              DrawDialog(gPrintDialog);
37 2003 -2      END
37 2004 --      ($IFC qDebug)
37 2005 --      ELSE
37 2006 --          ProgramBreak('TStdPrintHandler.ShowDocBeingPrinted: Unable to load dialog.')
37 2007 --      ($ENDC)      { semicolon };
37 2008 -1      END
37 2009 --      ELSE
37 2010 --          BanishPrintDialog;
37 2011 -0 A  END;
37 2012 --
37 2013 --
37 2014 --      {$S MAREs}
37 2015 --      {-----+
37 2016 --      |   ShowsOnScreen   |
37 2017 --      +-----}
37 2018 -- A  FUNCTION TStdPrintHandler.ShowsOnScreen: BOOLEAN;
37 2019 0- A  BEGIN
37 2020 --      IF fView <> NIL THEN
37 2021 --          ShowsOnScreen := fView.GetWindow <> NIL
37 2022 --      ELSE
37 2023 --          ShowsOnScreen := FALSE;
37 2024 -0 A  END;
37 2025 --
37 2026 --
37 2027 --      {$S PrintRes}
37 2028 --      {-----+
37 2029 --      |   StripToPage   |
37 2030 --      +-----}
37 2031 -- A  FUNCTION TStdPrintHandler.StripToPage (hStrip, vStrip: INTEGER): INTEGER;
37 2032 --      VAR
37 2033 --          strip:      Point;
37 2034 --          ortho:      VHSelect;
37 2035 --      BEGIN
37 2036 0- A  BEGIN
37 2037 --          SetPt(strip, hStrip, vStrip);
37 2038 --          ortho := gOrthogonal[fPageDirection];
37 2039 --          StripToPage := strip.vh[fPageDirection] * fPageStrips.vh[ortho] +
37 2040 --              strip.vh[ortho] + fStartPage;
37 2041 -0 A  END;
37 2042 --
37 2043 --
37 2044 --      {$S PrintTerminate}
37 2045 --      {-----+
37 2046 --      |   Terminate   |
37 2047 --      +-----}
37 2048 -- A  PROCEDURE TStdPrintHandler.Terminate; OVERRIDE;
37 2049 0- A  BEGIN
37 2050 --      EmpowerChooser(TRUE); { reenable Chooser, in case it was disabled }
37 2051 -0 A  END;
37 2052 --
37 2053 --
37 2054 --      {$S PrintRes}
37 2055 --      {-----+
37 2056 --      |   ValidatePrintRecord   |
37 2057 --      +-----}
37 2058 -- A  PROCEDURE TStdPrintHandler.ValidatePrintRecord (VAR didChange: BOOLEAN);
37 2059 --      VAR
37 2060 --          fi:      FailInfo;
37 2061 --      BEGIN
37 2062 --          {-----+
37 2063 --          |   CallPrValidate   |
37 2064 --          +-----}
37 2065 -- B  PROCEDURE CallPrValidate;
37 2066 0- B  BEGIN
37 2067 --          didChange := PrValidate(THPrint(fHPrint));
37 2068 -0 B  END;
37 2069 --
37 2070 --      {-----+
37 2071 --      |   Hd1ValidateFailed   |
37 2072 --      +-----}
37 2073 -- B  PROCEDURE Hd1ValidateFailed (error: OSErr; message: LONGINT);
37 2074 0- B  BEGIN
37 2075 --          Reset;
37 2076 --          EXIT(ValidatePrintRecord);
37 2077 -0 B  END;
37 2078 --

```

```

37 2079 0- A BEGIN
37 2080 -- CatchFailures(fi, Hd1ValidateFailed);
37 2081 -- DoInMacPrint(CallPrValidate);
37 2082 -- Success(fi);
37 2083 -0 A END;
37 2084 --
37 2085 -- (*****
37 2086 -- (* T P r i n t S t y l e C h a n g e C o m m a n d *)
37 2087 -- (*****
37 2088 -- {$$ PrintDoCommand}
37 2089 -- {-----+
37 2090 -- | IPrintStyleChangeCommand |
37 2091 -- +-----}
37 2092 -- A PROCEDURE TPrintStyleChangeCommand.IPrintStyleChangeCommand(itsPrintHandler: TStdPrintHandler);
37 2093 -- VAR
37 2094 -- fi: FailInfo;
37 2095 --
37 2096 -- {-----+
37 2097 -- | Hd1InitFailed |
37 2098 -- +-----}
37 2099 -- B PROCEDURE Hd1InitFailed(error: OSErr; message: LONGINT);
37 2100 0- B BEGIN
37 2101 -- Free;
37 2102 -0 B END;
37 2103 --
37 2104 0- A BEGIN
37 2105 -- fStdPrintHandler := itsPrintHandler;
37 2106 -- fOldHPrint := NIL;
37 2107 -- fNewHPrint := NIL;
37 2108 -- ICommand(cChangePrinterStyle, itsPrintHandler.fDocument, NIL, NIL);
37 2109 -- CatchFailures(fi, Hd1InitFailed);
37 2110 -- fOldHPrint := NewPermHandle(SIZEOF(TPrint));
37 2111 -- FailNIL(fOldHPrint);
37 2112 -- { Make a copy of the old version of the PrintInfo record }
37 2113 -- BlockMove(itsPrintHandler.fHPrint^, fOldHPrint^, SIZEOF(TPrint));
37 2114 -- fNewHPrint := NewPermHandle(SIZEOF(TPrint));
37 2115 -- FailNIL(fNewHPrint);
37 2116 -- Success(fi);
37 2117 -0 A END;
37 2118 --
37 2119 --
37 2120 -- {$$ PrintDoCommand}
37 2121 -- {-----+
37 2122 -- | Free |
37 2123 -- +-----}
37 2124 -- A PROCEDURE TPrintStyleChangeCommand.Free; OVERRIDE;
37 2125 0- A BEGIN
37 2126 -- IF fOldHPrint <> NIL THEN
37 2127 -- DisposHandle(fOldHPrint);
37 2128 -- IF fNewHPrint <> NIL THEN
37 2129 -- DisposHandle(fNewHPrint);
37 2130 -- INHERITED Free;
37 2131 -0 A END;
37 2132 --
37 2133 --
37 2134 -- {$$ PrintDoCommand}
37 2135 -- {-----+
37 2136 -- | DoIt |
37 2137 -- +-----}
37 2138 -- A PROCEDURE TPrintStyleChangeCommand.DoIt; OVERRIDE;
37 2139 0- A BEGIN
37 2140 -- fStdPrintHandler.CheckPrinter; { Will find it changed, and hence dispatch to view's DoPrinterChanged }
37 2141 -0 A END;
37 2142 --
37 2143 --
37 2144 -- {$IFC qDebug}
37 2145 -- {$$ PrintInspect}
37 2146 -- {-----+
37 2147 -- | Fields |
37 2148 -- +-----}
37 2149 -- A PROCEDURE TPrintStyleChangeCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
37 2150 -- fieldAddr: Ptr;
37 2151 -- fieldType: INTEGER));
37 2152 0- A BEGIN
37 2153 -- DoToField('fStdPrintHandler', @fStdPrintHandler, bObject);
37 2154 -- DoToField('fOldHPrint', @fOldHPrint, bHandle);
37 2155 -- DoToField('fNewHPrint', @fNewHPrint, bHandle);
37 2156 -- INHERITED Fields(DoToField);
37 2157 -0 A END;
37 2158 -- {$ENDC}
37 2159 --
37 2160 --
37 2161 -- {$$ PrintDoCommand}
37 2162 -- {-----+
37 2163 -- | UndoIt |
37 2164 -- +-----}
37 2165 -- A PROCEDURE TPrintStyleChangeCommand.UndoIt; OVERRIDE;
37 2166 0- A BEGIN
37 2167 -- BlockMove(fOldHPrint^, fStdPrintHandler.fHPrint^, SIZEOF(TPrint));
37 2168 -- fStdPrintHandler.CheckPrinter; { Will find it changed, and hence dispatch to
37 2169 -- view's DoPrinterChanged }
37 2170 -0 A END;
37 2171 --
37 2172 --
37 2173 -- {$$ PrintDoCommand}
37 2174 -- {-----+
37 2175 -- | RedoIt |
37 2176 -- +-----}
37 2177 -- A PROCEDURE TPrintStyleChangeCommand.RedoIt; OVERRIDE;

```

```
37 2178 0- A BEGIN
37 2179 --      BlockMove(fNewHPrint^, fStdPrintHandler.fHPrint^, SIZEOF(TPrint));
37 2180 --      fStdPrintHandler.CheckPrinter; { Will find it changed, and hence dispatch to
37 2181 --                               view's DoPrinterChanged }
37 2182 -0 A END;
37 2183 --
36 549 --
36 550 --      END.
```

```

38 1 -- {SP}
38 2 -- { UTEView.p }
38 3 -- { Copyright © 1986-1988 Apple Computer, Inc. All rights reserved. }
38 4 --
38 5 -- UNIT UTEView;
38 6 --
38 7 -- (*****
38 8 -- INTERFACE
38 9 -- (*****
38 10 --
38 11 -- {-----+
38 12 -- |   Flags   |
38 13 -- +-----+
38 14 -- {SSC+} { Use short circuit ops! }
38 15 --
38 16 -- {-----+
38 17 -- |   Uses   |
38 18 -- +-----+
38 19 -- USES
38 20 --     {$LOAD MacIntf.LOAD}
38 21 --     MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
38 22 --     {$LOAD UMacApp.LOAD}
38 23 --     UMAUtil, UViewCoords, UFailure,
38 24 --     UMemory, UMenuSetup, UObject, UList, UAssociation, UMacApp,
38 25 --     {$LOAD}
38 26 --     {$IFC qDebug} UWriteLnWindow, {$ENDC}
38 27 --     Script;
38 28 --
38 29 -- {$IFC qMPW2} { TextEdit interface that isn't included in MPW 2.x. }
38 30 -- CONST
38 31 --     doToggle = 32;
38 32 --
38 33 -- TYPE
38 34 --     TEIntHook = (intEOLHook, intDrawHook, intWidthHook, intHitTestHook);
38 35 --
38 36 -- FUNCTION TEContinuousStyle(VAR mode: INTEGER; VAR aStyle: TextStyle;
38 37 --     hTE: TEHandle): BOOLEAN;
38 38 --     INLINE $3F3C, $000A, $A83D;
38 39 --
38 40 -- PROCEDURE SetStylScrap(rangeStart, rangeEnd: LONGINT; newStyles: StScrpHandle;
38 41 --     redraw: BOOLEAN; hTE: TEHandle);
38 42 --     INLINE $3F3C, $000B, $A83D;
38 43 --
38 44 -- PROCEDURE TECustomHook(which: TEIntHook; VAR addr: ProcPtr; hTE: TEHandle);
38 45 --     INLINE $3F3C, $000C, $A83D;
38 46 --
38 47 -- FUNCTION TEnumStyles(rangeStart, rangeEnd: LONGINT; hTE: TEHandle): LONGINT;
38 48 --     INLINE $3F3C, $000D, $A83D;
38 49 -- {$ENDC qMPW2}
38 50 --
38 51 -- {-----+
38 52 -- |   global constants   |
38 53 -- +-----+
38 54 -- CONST
38 55 --     kUnlimited = MAXINT; { The maximum number of characters in the fText
38 56 --     of a TTEView object }
38 57 --
38 58 --     kWithStyle = TRUE; { Parameters to TTEView.ITEView }
38 59 --     kWithoutStyle = FALSE;
38 60 --     kSaveCurrentChars = TRUE; { Parameter to ITECommand }
38 61 --
38 62 -- {-----+
38 63 -- |   global types   |
38 64 -- +-----+
38 65 -- TYPE
38 66 -- {-----+
38 67 -- |   TTEView   |
38 68 -- +-----+
38 69 -- TTEViewTemplatePtr = ^TTEViewTemplate;
38 70 -- TTEViewTemplate = PACKED RECORD
38 71 --     itsStyleType: BOOLEAN;
38 72 --     itsAutoWrap: BOOLEAN;
38 73 --     itsAcceptsChanges: BOOLEAN;
38 74 --     itsFreesText: BOOLEAN;
38 75 --     filler1: 0..4095;
38 76 --     itsKeyCmdNumber: INTEGER;
38 77 --     itsMaxChars: INTEGER;
38 78 --     itsInset: Rect;
38 79 --     itsJustification: INTEGER;
38 80 --     itsTextFace: Style;
38 81 --     itsTextSize: INTEGER;
38 82 --     itsTextColor: RGBColor;
38 83 --     itsFontName: Str255; { Actually a variable length P-String }
38 84 -- END;
38 85 --
38 86 -- TTEView = OBJECT(TView)
38 87 --
38 88 -- { TTEView is a view subclass representing a TextEdit record. TextEdit is
38 89 -- the simple text-editing facility built into the Macintosh ROM. The
38 90 -- purpose of a TTEView is to make TextEdit function properly in a MacApp
38 91 -- environment, for things like scrolling, printing, page breaks, and
38 92 -- command handling }
38 93 --
38 94 -- { Fields }
38 95 --
38 96 --     fHTE: TEHandle; { Handle to the actual TextEdit object }
38 97 --
38 98 --     fText: Handle; { The text in the TEHandle }

```

```

38 99 --      fSavedTEHandle:    Handle;    { Saved handle from TENew }
38 100 --
38 101 --      fInset:              Rect;      { Amount to inset text from edges of view }
38 102 --
38 103 --      fKeyCmdNumber:      CmdNumber; { Will be used as the string number for
38 104 --                  "Undo Typing" }
38 105 --
38 106 --      fMaxChars:          INTEGER;   { Maximum number of chars to accept into
38 107 --                  fText. Default is MAXINT; stuff to
38 108 --                  different value if you want to
38 109 --                  limit max chars to fewer than
38 110 --                  MAXINT }
38 111 --
38 112 --      fLastHeight:        INTEGER;   { Last checked height of record. If the record is
38 113 --                  stylish, this represents the height in pixels.
38 114 --                  For old-style records, equals number of lines. }
38 115 --
38 116 --      fTypingCommand:     TTETypingCommand;
38 117 --                  { The current TE Typing command relating
38 118 --                  to me, if any }
38 119 --
38 120 --      fTextStyle:        TextStyle; { Current style of text }
38 121 --
38 122 --      fJustification:     INTEGER;   { Current justification of text record }
38 123 --
38 124 --      fAcceptsChanges:    BOOLEAN;   { Set to FALSE to have text which will
38 125 --                  not accept any change, such as text
38 126 --                  on the Clipboard, or perhaps
38 127 --                  received mail }
38 128 --      fStyleType:        BOOLEAN;   { Set to kWithStyle if record is styled }
38 129 --      fAutoWrap:          BOOLEAN;   { Set to FALSE for line wrapping at CR's only }
38 130 --      fFreeText:          BOOLEAN;   { Determines if fText should be freed on
38 131 --                  Free. }
38 132 --      fSpecsChanged:      BOOLEAN;   { Something recently happened which could effect
38 133 --                  font/style/size/color menu item updating.
38 134 --                  Should be reset to FALSE when application
38 135 --                  has taken the appropriate action. }
38 136 --      fScroller:          TScroller; { The scroller in which this view is contained }
38 137 --      fLastPageBreak:     INTEGER;   { Caches last page break computed. }
38 138 --      fLastLine:          INTEGER;   { Last line of text of the last page break }
38 139 -- { Initialize and Free }
38 140 --
38 141 --      PROCEDURE TTEView.ITEView(
38 142 --          itsDocument:      TDocument; { OK to be NIL, if view will
38 143 --                  belong to a documentless
38 144 --                  window }
38 145 --          itsSuperview:     TView;      { The view in which this view is contained }
38 146 --          itsLocation:      VPoint;     { Location in its superview }
38 147 --          itsSize:          VPoint;     { Size of view }
38 148 --          itsHDeterminer:   SizeDeterminer; { How to width of the view is to
38 149 --                  be determined }
38 150 --          itsVDeterminer:   SizeDeterminer; { How the height of the view is
38 151 --                  to be determined }
38 152 --          itsInset:         Rect;      { Amount to inset text }
38 153 --          itsTextStyle:     TextStyle; { Initial text style }
38 154 --          itsJustification: INTEGER;     { Its justification }
38 155 --          itsStyleType:     BOOLEAN;   { Whether or not record is styled }
38 156 --          itsAutoWrap:      BOOLEAN    { FALSE if newline occurs at CR only }
38 157 --      );
38 158 --
38 159 -- { $IFC qTemplateViews }
38 160 --      PROCEDURE TTEView.IRes (itsDocument: TDocument; itsSuperview: TView; VAR itsParams: Ptr); OVERRIDE;
38 161 --          { Initialize a TTEView via a 'view' resource. }
38 162 -- { $ENDC }
38 163 --
38 164 -- { $IFC qWriteTemplates }
38 165 --      PROCEDURE TTEView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
38 166 --
38 167 --      PROCEDURE TTEView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
38 168 -- { $ENDC }
38 169 --
38 170 --      PROCEDURE TTEView.Free; OVERRIDE;
38 171 --          { Frees the TRecord and optionally fText, then frees SELF }
38 172 --
38 173 --      PROCEDURE TTEView.MakeTRecord;
38 174 --          { Called with grafport properly set, to create the actual TE Record }
38 175 --
38 176 -- { Commands and Menus }
38 177 --
38 178 --      FUNCTION TTEView.DoKeyCommand(ch: Char; aKeyCode: INTEGER;
38 179 --          VAR info: EventInfo): TCommand; OVERRIDE;
38 180 --          { Handles keystrokes }
38 181 --
38 182 --      FUNCTION TTEView.DoMakeEditCommand(aCmdNumber: CmdNumber): TTECommand;
38 183 --          { Make a command for handling edit menu stuff }
38 184 --
38 185 --      FUNCTION TTEView.DoMakeStyleCommand(aStyle: TextStyle; itsCmdNumber: CmdNumber;
38 186 --          itsMode: INTEGER): TTEStyleCommand;
38 187 --          { Make a style change command }
38 188 --
38 189 --      FUNCTION TTEView.DoMakeTypingCommand(ch: Char): TTETypingCommand;
38 190 --          { Make a typing command for handling keystrokes }
38 191 --
38 192 --      FUNCTION TTEView.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
38 193 --          { Handles menu commands }
38 194 --
38 195 --      FUNCTION TTEView.DoMouseCommand(VAR theMouse: Point;
38 196 --          VAR info: EventInfo; VAR hysteresis: Point): TCommand; OVERRIDE;
38 197 --          { Handles mousepresses }

```

```

38 198 --
38 199 --      PROCEDURE TTEView.DoneTyping;
38 200 --          { No further typing can occur for the current typing command }
38 201 --
38 202 --      PROCEDURE TTEView.DoSetupMenus; OVERRIDE;
38 203 --          { Sets up menus }
38 204 --
38 205 --      PROCEDURE TTEView.InstallSelection(wasActive, beActive: BOOLEAN); OVERRIDE;
38 206 --          { Called at activate/deactivate time to get insertion-point-blinking
38 207 --            right (or to highlight the selection) }
38 208 --
38 209 --      { Screen Display }
38 210 --
38 211 --      PROCEDURE TTEView.DoIdle(phase: IdlePhase); OVERRIDE;
38 212 --          { Blinks the insertion point }
38 213 --
38 214 --      FUNCTION TTEView.DoSetCursor(localPoint: Point;
38 215 --          cursorRgn: RgnHandle): BOOLEAN; OVERRIDE;
38 216 --          { Sets the cursor to the I-beam }
38 217 --
38 218 --      PROCEDURE TTEView.Draw(area: Rect); OVERRIDE;
38 219 --          { Draw the text in a frame or on the printed page }
38 220 --
38 221 --      PROCEDURE TTEView.ShowReverted; OVERRIDE;
38 222 --          { Make sure line starts are correct before redisplay }
38 223 --
38 224 --
38 225 --      { Size Changes }
38 226 --
38 227 --      PROCEDURE TTEView.BeInPort (itsPort: GrafPtr); OVERRIDE;
38 228 --          { Now that a grafport is established for the view, tells TextEdit about it }
38 229 --
38 230 --      PROCEDURE TTEView.BeInScroller (itsScroller: TScroller); OVERRIDE;
38 231 --          { Gives us a chance to set the scroll parameters }
38 232 --
38 233 --      PROCEDURE TTEView.CalcMinSize(VAR minSize: VPoint); OVERRIDE;
38 234 --          { Compute the minimum size of the view }
38 235 --
38 236 --      FUNCTION TTEView.CalcRealHeight: INTEGER;
38 237 --          { Calculate true height of record, including last character if it is a
38 238 --            carriage return }
38 239 --
38 240 --      PROCEDURE TTEView.ComputeSize(VAR newSize: VPoint); OVERRIDE;
38 241 --          { Compute the actual size of the view }
38 242 --
38 243 --      PROCEDURE TTEView.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
38 244 --          { Installs the new size for both MacApp and TextEdit }
38 245 --
38 246 --      PROCEDURE TTEView.SynchView (redraw: BOOLEAN);
38 247 --          { Keeps view metrics in synch after TextEdit operation. If redraw is
38 248 --            true then ScrollSelectionIntoView is called to make sure the
38 249 --            selection is visible.}
38 250 --
38 251 --      { Clipboard }
38 252 --
38 253 --          { NOTE -- These methods apply only to a TTEView installed as the view in
38 254 --            the Clipboard }
38 255 --
38 256 --      FUNCTION TTEView.ContainsClipType(aType: ResType): BOOLEAN; OVERRIDE;
38 257 --          { Determines whether the indicated clipboard type can be produced by the view }
38 258 --
38 259 --      FUNCTION TTEView.GivePasteData(aDataHandle: Handle; dataType: ResType): LONGINT; OVERRIDE;
38 260 --          { Supplies data to be PASTED by some requesting PASTE command }
38 261 --
38 262 --      PROCEDURE TTEView.WriteToDeskScrap; OVERRIDE;
38 263 --          { Produces TEXT data for the Desk Scrap when called upon to do so }
38 264 --
38 265 --      { Miscellaneous }
38 266 --
38 267 --      PROCEDURE TTEView.AutoScrolling (doScrolling: BOOLEAN);
38 268 --          { Set TRecord auto scrolling to given value via TEAutoScroll }
38 269 --
38 270 --      PROCEDURE TTEView.CalcSelLoc(VAR selectionRect: Rect);
38 271 --          { Calculates the location of the current Selection }
38 272 --
38 273 --      PROCEDURE TTEView.ChangeWrap (newAutoWrap, redraw: BOOLEAN);
38 274 --          { Changes auto-wrapping behavior, redrawing if requested }
38 275 --
38 276 --      FUNCTION TTEView.ContinuousStyle(firstChar, lastChar: INTEGER;
38 277 --          VAR mode: INTEGER; VAR aStyle: TextStyle): BOOLEAN;
38 278 --          { Returns TRUE if the style of the characters within the given range is
38 279 --            wholly continuous. The style attributes to check are specified by mode,
38 280 --            which has the same meaning as in the TextEdit call TSetStyle. Attributes
38 281 --            which are continuous are returned in aStyle, with the appropriate bits set
38 282 --            in mode. Font style attribute bits are individually checked across
38 283 --            the selection range, and returned set to 1 if that attribute is continuous. }
38 284 --
38 285 --      PROCEDURE TTEView.ExtractStyles(VAR theStyles: TStyleHandle; VAR theElements: STHandle);
38 286 --          { Extract handles to style information for a TextEdit record, presumably
38 287 --            for the purpose of storing the data to disk. Note that line heights
38 288 --            can be recalculated, and as such don't need to be stored. }
38 289 --
38 290 --      FUNCTION TTEView.ExtractText: Handle;
38 291 --          { For sake of completeness. Extract handle to text, presumably for the
38 292 --            purpose of storing the text to disk. }
38 293 --
38 294 --      PROCEDURE TTEView.RecalcText;
38 295 --          { Tells TextEdit to recompute linestarts }
38 296 --

```

```

38 297 --      PROCEDURE TTEView.ScrollSelectionIntoView;
38 298 --          { Scrolls the selection into view -- called after Commands are done or undone
38 299 --            If auto scrolling is on, calls TESelView to do the right thing }
38 300 --
38 301 --      PROCEDURE TTEView.SetJustification(newJust: INTEGER; redraw: BOOLEAN);
38 302 --          { Sets the TEREcord to the given justification. If redraw is true then
38 303 --            the view is redrawn by calling ForceRedraw. Otherwise it is assumed
38 304 --            the called will get the view redrawn. }
38 305 --
38 306 --      PROCEDURE TTEView.SetOneStyle(theStart, theEnd, theMode: INTEGER;
38 307 --          theStyle: TextStyle; redraw: BOOLEAN);
38 308 --          { Sets a text style continuously over a range. In unstylish TERERecords,
38 309 --            the style is set across the entire record. If redraw is true then
38 310 --            the text is redrawn. }
38 311 --
38 312 --      PROCEDURE TTEView.SetText(theText: Str255);
38 313 --          { Convenience routine, to save from having to create a handle just to pass
38 314 --            to StuffText. Note that if fSavedTEHandle already holds a saved handle,
38 315 --            it will be disposed before StuffText is called. }
38 316 --
38 317 --      FUNCTION TTEView.SpaceForStyles(rangeStart, rangeEnd: LONGINT): BOOLEAN;
38 318 --          { Returns TRUE if there is enough memory to hold the styles over the
38 319 --            given range in the TEREcord }
38 320 --
38 321 --      PROCEDURE TTEView.StuffStyles(theStyles: TStyleHandle; theElements: STHandle);
38 322 --          { Installs style information, presumably fetched from disk, into the
38 323 --            TextEdit record. }
38 324 --
38 325 --      PROCEDURE TTEView.StuffText(theText: Handle);
38 326 --          { Installs text into the TextEdit record. Note that this destroys any
38 327 --            existing style information -- StuffStyles should be called if
38 328 --            styles are to be attached to this text information }
38 329 --
38 330 --      PROCEDURE TTEView.StuffTERects(newTERect: Rect);
38 331 --          { installs "newTERect" as the dest & view rect of the TextEdit record }
38 332 --
38 333 --      { Printing }
38 334 --
38 335 --      FUNCTION TTEView.DoBreakFollowing(vhs: VHSelect; prevBreak: VCoordinate;
38 336 --          VAR automatic: BOOLEAN): VCoordinate; OVERRIDE;
38 337 --          { Computes the next page-break following 'prevBreak' in direction
38 338 --            given by 'vhs' }
38 339 --
38 340 --      PROCEDURE TTEView.DoCalcViewPerPage (VAR viewPerPage: VPoint); OVERRIDE;
38 341 --          { Computes how much of the view is to be allocated to each printed page }
38 342 --
38 343 --      PROCEDURE TTEView.DoSetPageOffset(coord: VPoint); OVERRIDE;
38 344 --          { At printing time, determine the location of the 'interior' (body)
38 345 --            of the page }
38 346 --
38 347 --      PROCEDURE TTEView.GetPrintExtent (VAR printExtent: VRect); OVERRIDE;
38 348 --          { Print extent is the view's extent minus the insets. }
38 349 --
38 350 --      { Debugging }
38 351 --
38 352 --      ($IFC qDebug)
38 353 --          PROCEDURE TTEView.IdentifySoftware; OVERRIDE;
38 354 --              { Tells compile date of this unit }
38 355 --
38 356 --          PROCEDURE TTEView.Fields (PROCEDURE DoToField (fieldName: Str255;
38 357 --              fieldAddr: Ptr;
38 358 --              fieldType: INTEGER)); OVERRIDE;
38 359 --              { Tell interesting stuff }
38 360 --      ($ENDC)
38 361 --      END;
38 362 --
38 363 --
38 364 --      { A command that adds characters to, or delete characters from, a TTEView }
38 365 --      {-----+
38 366 --      | TTECommand |
38 367 --      +-----+}
38 368 --      TTECommand = OBJECT(TCommand)
38 369 --
38 370 --      { Fields }
38 371 --
38 372 --      fTEView:    TTEView;          { The TEView operated on }
38 373 --
38 374 --      fHTE:       TEHandle;          { same as fTEView's fHTE; duplicated for code
38 375 --                                     efficiency }
38 376 --
38 377 --      fOldStart:  INTEGER;           { The beginning and ending positions of the }
38 378 --      fOldEnd:    INTEGER;           { selection at the moment just before the
38 379 --                                     command was done }
38 380 --
38 381 --      fOldText:   Handle;            { If fOldStart = fOldEnd, i.e., if old selection
38 382 --                                     had been an insertion point, this will be
38 383 --                                     NIL. Otherwise, provides a temporary home
38 384 --                                     for the characters comprising the old
38 385 --                                     selection }
38 386 --
38 387 --      fOldStyles: STScrpHandle;
38 388 --
38 389 --      fNewStart:  INTEGER;           { The beginning and ending locations in the Text... }
38 390 --      fNewEnd:    INTEGER;           { _of the new text that is added by the
38 391 --                                     command, if any }
38 392 --
38 393 --      fNewText:   Handle;            { A Handle to the characters added by the command }
38 394 --      fNewStyles: STScrpHandle;
38 395 --
38 396 --      fPadding:   Handle;            { Handle to fill size between new and old. This
38 397 --                                     insures that we can always undo and redo. }
38 398 --      fTextPad:   INTEGER;           { Size difference between New and Old text }

```



```

38 396 --      fStylePad: LONGINT;          { Size difference between New and Old styles }
38 397 --
38 398 --      { Initialize and Free }
38 399 --
38 400 --      PROCEDURE TTECommand.ITECommand(itsTEView: TTEView; itsCmdNumber: CmdNumber; itsSaveText: BOOLEAN);
38 401 --      { Initialize the command; if unsuccessful, exit is via Failure mechanism. }
38 402 --
38 403 --      PROCEDURE TTECommand.Free; OVERRIDE;
38 404 --      { Free the text handles holding information needed for Undo/Redo, then Frees SELF }
38 405 --
38 406 --      { Command execution phase overrides }
38 407 --
38 408 --      PROCEDURE TTECommand.DoIt; OVERRIDE;
38 409 --      { Focuses, then calls DoMainFunction }
38 410 --
38 411 --      PROCEDURE TTECommand.RedoIt; OVERRIDE;
38 412 --      { Focuses, calls RestoreSelection to get selection right, then calls DoMainFunction
38 413 --        to reinstate the changes done in DoIt which were undone by a preceding UndoIt }
38 414 --
38 415 --      PROCEDURE TTECommand.UndoIt; OVERRIDE;
38 416 --      { Focuses, then dispatches to RemoveAdditions, ReviveDeletions,
38 417 --        and RestoreSelection to accomplish the Undo }
38 418 --
38 419 --      { Command execution - Restoration of Selection -- called in both UNDO and REDO phases }
38 420 --
38 421 --      PROCEDURE TTECommand.RestoreSelection;
38 422 --      { Set the Selection to be what it was just before the DO phase of
38 423 --        the command was performed }
38 424 --
38 425 --      { Command execution - steps in DO and REDO phases }
38 426 --
38 427 --      PROCEDURE TTECommand.BanishOldText;
38 428 --      { Remove text that was selected at the outset of the Do phase }
38 429 --
38 430 --      PROCEDURE TTECommand.InstallNewText;
38 431 --      { Install the new text }
38 432 --
38 433 --      PROCEDURE TTECommand.DoMainFunction;
38 434 --      { Dispatches to the relevant methods for Do and Redo phases }
38 435 --
38 436 --      { Command execution - steps in UNDO phase }
38 437 --
38 438 --      PROCEDURE TTECommand.RemoveAdditions;
38 439 --      { Remove any characters which were added by the DO phase of the
38 440 --        command }
38 441 --
38 442 --      PROCEDURE TTECommand.ReviveDeletions;
38 443 --      { Bring back the characters which were removed during the DO phase }
38 444 --
38 445 --      {$IFC qDebug}
38 446 --      { Debugging }
38 447 --
38 448 --      PROCEDURE TTECommand.Fields (PROCEDURE DoToField (fieldName: Str255;
38 449 --                                           fieldAddr: Ptr;
38 450 --                                           fieldType: INTEGER)); OVERRIDE;
38 451 --      {$ENDC}
38 452 --
38 453 --      END;
38 454 --
38 455 --
38 456 --      {-----+
38 457 --      | TTECutCopyCommand |
38 458 --      +-----+
38 459 --      TTECutCopyCommand = OBJECT(TTECommand)
38 460 --
38 461 --      fClipCreated: BOOLEAN;          { Clipboard view created OK }
38 462 --
38 463 --      { Creation and Destruction }
38 464 --
38 465 --      PROCEDURE TTECutCopyCommand.ITECutCopyCommand(itsTEView: TTEView;
38 466 --                                                    itsCmdNumber: CmdNumber);
38 467 --      { Initializes the command }
38 468 --
38 469 --      PROCEDURE TTECutCopyCommand.Free; OVERRIDE;
38 470 --      { Free the command }
38 471 --
38 472 --      { Command execution }
38 473 --
38 474 --      PROCEDURE TTECutCopyCommand.DoIt; OVERRIDE;
38 475 --      { Launches a TEView for installation in the Clipboard, then calls
38 476 --        DoMainFunction }
38 477 --
38 478 --      PROCEDURE TTECutCopyCommand.ReviveDeletions; OVERRIDE;
38 479 --      { Bring back the characters which were removed during the DO phase;
38 480 --        called during UNDO phase }
38 481 --
38 482 --      END;
38 483 --
38 484 --
38 485 --      {-----+
38 486 --      | TTEPasteCommand |
38 487 --      +-----+
38 488 --      TTEPasteCommand = OBJECT(TTECommand)
38 489 --
38 490 --      { Initialization }
38 491 --
38 492 --      PROCEDURE TTEPasteCommand.ITEPasteCommand(itsTEView: TTEView);
38 493 --      { Initialize the command; if unsuccessful, signalled by Failure mechanism }
38 494 --

```

```

38 495 --      END;
38 496 --
38 497 --
38 498 --      {-----+
38 499 --      | TTEStyleCommand |
38 500 --      +-----+
38 501 --      TTEStyleCommand = OBJECT(TTECommand)
38 502 --
38 503 --      { Fields }
38 504 --      { These two fields are only used in non-styled TextEdit records }
38 505 --      fMode:          INTEGER;          { Mode for style change }
38 506 --      fOldTextStyle:  TextStyle;        { The original text style }
38 507 --      fNewTextStyle:  TextStyle;        { What we're replacing it with }
38 508 --
38 509 --      { Initialization }
38 510 --
38 511 --      PROCEDURE TTEStyleCommand.ITESTyleCommand(itsTEView: TTEView; itsNewStyle: TextStyle;
38 512 --      itsCmdNumber: CmdNumber; itsMode: INTEGER);
38 513 --      { Initialize the command; if unsuccessful, signalled by Failure mechanism }
38 514 --
38 515 --      { Command execution }
38 516 --
38 517 --      PROCEDURE TTEStyleCommand.InstallOneStyle(newStyl: TextStyle);
38 518 --      { Installs one style over entire record - for use with old TextEdit }
38 519 --
38 520 --      PROCEDURE TTEStyleCommand.InstallManyStyles(newStyls: StScrpHandle);
38 521 --      { Installs styles over selection range - for use with styled TextEdit }
38 522 --
38 523 --      PROCEDURE TTEStyleCommand.DoIt; OVERRIDE;
38 524 --      { Applies style to selection range }
38 525 --
38 526 --      PROCEDURE TTEStyleCommand.UndoIt; OVERRIDE;
38 527 --
38 528 --      PROCEDURE TTEStyleCommand.RedoIt; OVERRIDE;
38 529 --
38 530 --      {IFC qDebug}
38 531 --      { Debugging }
38 532 --      PROCEDURE TTEStyleCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
38 533 --      fieldAddr: Ptr;
38 534 --      fieldType: INTEGER)); OVERRIDE;
38 535 --
38 536 --      {SENDC}
38 537 --
38 538 --      END;
38 539 --
38 540 --      {-----+
38 541 --      | TTETypingCommand |
38 542 --      +-----+
38 543 --      TTETypingCommand = OBJECT(TTECommand)
38 544 --
38 545 --      fCompleted: BOOLEAN; { Indicates whether further keystrokes will be
38 546 --      extensions to this command (FALSE) or
38 547 --      whether the command has already been
38 548 --      completed (TRUE) }
38 549 --      fFirstChar: CHAR;    { First character typed }
38 550 --
38 551 --      { Creation and destruction }
38 552 --
38 553 --      PROCEDURE TTETypingCommand.ITETypingCommand(itsTEView: TTEView;
38 554 --      itsFirstChar: Char);
38 555 --      { Initialize the command; if not successful, exit is via Failure
38 556 --      mechanism }
38 557 --
38 558 --      PROCEDURE TTETypingCommand.Free; OVERRIDE;
38 559 --      { Deallocate the command and dependent structures }
38 560 --
38 561 --      { Command processing }
38 562 --
38 563 --      PROCEDURE TTETypingCommand.DoNormalChar(aChar: CHAR);
38 564 --      { Handle any typed character except a backspace }
38 565 --
38 566 --      PROCEDURE TTETypingCommand.BkSpcLeft(theText: Handle; curStart: INTEGER);
38 567 --      { Handle backspace to the left of the original selection }
38 568 --
38 569 --      PROCEDURE TTETypingCommand.BkSpcRight(theText: Handle; curStart: INTEGER);
38 570 --      { Handle backspace to the right of the original selection }
38 571 --
38 572 --      PROCEDURE TTETypingCommand.AddCharacter(aChar: Char);
38 573 --      { Add another character to an already-launched TTETypingCommand }
38 574 --
38 575 --      PROCEDURE TTETypingCommand.DoIt; OVERRIDE;
38 576 --      { First processing for command; adds first character }
38 577 --
38 578 --      PROCEDURE TTETypingCommand.UndoIt; OVERRIDE;
38 579 --      { Process undo typing }
38 580 --
38 581 --      PROCEDURE TTETypingCommand.CompleteTyping;
38 582 --      { Mark as no more typing allowed and fix up style scrap, if any }
38 583 --
38 584 --      { Debugging }
38 585 --
38 586 --      PROCEDURE TTETypingCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
38 587 --      fieldAddr: Ptr;
38 588 --      fieldType: INTEGER)); OVERRIDE;
38 589 --
38 590 --      {SENDC}
38 591 --
38 592 --      END;
38 593 --      {-----+

```

```
38 594 -- | Global variables |
38 595 -- +-----+
38 596 -- VAR
38 597 --     defClikLoopProc:   ProcPtr;
38 598 --
38 599 -- {-----+
38 600 -- | Global routines |
38 601 -- +-----+
38 602 -- PROCEDURE InitUTEView;
38 603 --     { Initialize TEView unit. }
38 604 --
38 605 -- PROCEDURE SetSelect(theStart, theEnd: INTEGER; hTE: TEHandle);
38 606 --     { For those times when we don't want the selection range hilited when changed }
38 607 --
38 608 -- FUNCTION AutoscrollTEView: BOOLEAN;
38 609 --     { Called from TextEdit when the mouse is down, so that autoscrolling
38 610 --       may be done. }
38 611 --
38 612 -- {$IFC qDebug}
38 613 -- PROCEDURE DumpTERecord(aTEH: TEHandle);
38 614 --     { writes salient information about the TERecord out to the Debug window }
38 615 -- PROCEDURE DumpTTECommand(theTTECommand: TTECommand);
38 616 --     { writes TTECommand info out to Debug window }
38 617 -- {$ENDC}
38 618 --
38 619 -- (*****
38 620 -- IMPLEMENTATION
38 621 -- (*****
38 622 --
38 623 -- {$I UTEView.Globals.p}
```

```

39 1 --      {$F}
39 2 --      {UTableView.Globals.p}
39 3 --      { Copyright © 1985 - 1988 Apple Computer, Inc. All rights reserved. }
39 4 --
39 5 --      {-----+
39 6 --      | local variables
39 7 --      +-----}
39 8 --      VAR
39 9 --      gCurrTEView:    TTEView;                ( For the benefit of AutoScroll only )
39 10 --      {$IFC gDebug}
39 11 --      gTEIntenseDebugging:    BOOLEAN;
39 12 --      {$ENDC}
39 13 --
39 14 --      (*****
39 15 --      (*      U t i l i t y   r o u t i n e s
39 16 --      (*****
39 17 --      {$S TEInit}
39 18 --      {-----+
39 19 --      | InitUTableView
39 20 --      +-----}
39 21 -- A  PROCEDURE InitUTableView;
39 22 --      VAR
39 23 --      aTEView:    TTEView;
39 24 --
39 25 0- A  BEGIN
39 26 --      {$IFC qTemplateViews}
39 27 --      NEW(aTEView);      FailNIL(aTEView);      RegisterStdType('TTEView', 'tevv', aTEView);
39 28 --      {$ENDC}
39 29 0- A  END;
39 30 --
39 31 --      {$IFC gDebug}
39 32 --      {$S TEDebug}
39 33 --      {-----+      For debugging - prints out actual character at
39 34 --      | WriteChar      |      position "index" in text < 2047
39 35 --      +-----}
39 36 -- A  PROCEDURE WriteChar(index: INTEGER; hText: Handle);
39 37 --      CONST
39 38 --      kMaxIndex = 2047;
39 39 --      TYPE
39 40 --      x = PACKED ARRAY[0..kMaxIndex] OF Char;
39 41 --      PX = ^x;
39 42 --      HX = ^Px;
39 43 0- A  BEGIN
39 44 --      IF index <= kMaxIndex THEN
39 45 --      Write(['', index:1, ':', HX(hText)^[index:1, ' ]');
39 46 0- A  END;
39 47 --
39 48 --      {$S TEDebug}
39 49 --      {-----+      Reset kMaxCharsToPrint upwards as needed. Set
39 50 --      | DumpTERecord      |      to 100 to avoid runaway debug window output.
39 51 --      +-----}
39 52 -- A  PROCEDURE DumpTERecord(aTEH: TEHandle);
39 53 --      CONST
39 54 --      kMaxCharsToPrint = 100;
39 55 --      VAR
39 56 --      size:    INTEGER;
39 57 --      hText:    Handle;
39 58 --      i:    INTEGER;
39 59 --
39 60 0- A  BEGIN
39 61 --      Write('TE -- dest rect: '); WriteRect(aTEH^.destRect);
39 62 --      WriteLn;
39 63 --
39 64 --      size := aTEH^.teLength;
39 65 --      Write('Line ht: ', aTEH^.lineHeight:1);
39 66 --      Write('; teLength: ', size:1);
39 67 --      Write('; hText length: ', GetHandleSize(aTEH^.hText):1);
39 68 --      Write('; lines: ', aTEH^.nLines:1);
39 69 --      WriteLn;
39 70 --
39 71 --      IF gIntenseDebugging THEN
39 72 1-      BEGIN
39 73 --      hText := aTEH^.hText;
39 74 --      Write('Chars: ');
39 75 --      FOR i := 0 TO Min(kMaxCharsToPrint, Min(2047, size - 1)) DO
39 76 2-      BEGIN
39 77 --      WriteChar(i, hText);
39 78 --      IF i MOD 10 = 9 THEN
39 79 --      WriteLn;
39 80 2-      END;
39 81 --      WriteLn;
39 82 --
39 83 --      Write('InPort vis bbox: '); WriteRect(aTEH^.inPort^.visRgn^.rgnBBox);
39 84 --      Write('; clip bbox: '); WriteRect(aTEH^.inPort^.clipRgn^.rgnBBox); WriteLn;
39 85 --
39 86 --      IF aTEH^.inPort <> thePort THEN
39 87 2-      BEGIN
39 88 --      Write('thePort vis bbox: '); WriteRect(thePort^.visRgn^.rgnBBox);
39 89 --      Write('; clip bbox: '); WriteRect(thePort^.clipRgn^.rgnBBox); WriteLn;
39 90 2-      END;
39 91 1-      END;
39 92 --
39 93 0- A  END;
39 94 --
39 95 --
39 96 --      {$S TEFields}
39 97 --      {-----+
39 98 --      | DumpTTECommand      |      Debug routine to print out involved TTECommands

```

```

39 99 --      +-----}
39 100 -- A    PROCEDURE DumpTTECommand(theTTECommand: TTECommand);
39 101 --      TYPE
39 102 --          TSPtr = ^TextStyle;
39 103 --      VAR
39 104 --          i:          LONGINT;
39 105 --          oldStyleSize: LONGINT;
39 106 --          newStyleSize: LONGINT;
39 107 --          last:       LONGINT;
39 108 0- A    BEGIN
39 109 --          WITH theTTECommand DO
39 110 --              WRITELN('old start/end, new start/end, diff :',
39 111 --                  foldStart:0, '/', foldEnd:0, ', ',
39 112 --                  fNewStart:0, '/', fNewEnd:0, ', ',
39 113 --                  MAX(MAX(foldEnd-foldStart, 0), MAX(fNewEnd-fNewStart, 0)):0);
39 114 --              WRITELN('fTextPad :', theTTECommand.fTextPad:0);
39 115 --              WRITELN('fPadding size :', GetHandleSize(theTTECommand.fPadding):0);
39 116 --
39 117 --              oldStyleSize := 0;
39 118 --              newStyleSize := 0;
39 119 --              WRITELN('** Old styles:');
39 120 --              IF theTTECommand.foldStyles = NIL THEN
39 121 --                  WRITELN('NONE!')
39 122 --              ELSE
39 123 1-          BEGIN
39 124 --              last := theTTECommand.foldStyles^.scrpNStyles;
39 125 --              WRITELN('Number of table entries: ', last:0);
39 126 --              FOR i := 0 TO last-1 DO
39 127 --                  WITH theTTECommand.foldStyles^.scrpStyleTab[i] DO
39 128 2-              BEGIN
39 129 --                  WRITE(i:0, ', ofs:', scrpStartChar:0, ' ');
39 130 --                  WritelnField(' ', @scrpFont, bStyle);
39 131 --                  WRITELN;
39 132 -2              END;
39 133 --                  oldStyleSize := GetHandleSize(Handle(theTTECommand.foldStyles));
39 134 -1              END;
39 135 --
39 136 --              WRITELN('** New styles:');
39 137 --              IF theTTECommand.fNewStyles = NIL THEN
39 138 --                  WRITELN('NONE!')
39 139 --              ELSE
39 140 1-          BEGIN
39 141 --              last := theTTECommand.fNewStyles^.scrpNStyles;
39 142 --              WRITELN('Number of table entries: ', last:0);
39 143 --              FOR i := 0 TO last-1 DO
39 144 --                  WITH theTTECommand.fNewStyles^.scrpStyleTab[i] DO
39 145 2-              BEGIN
39 146 --                  WRITE(i:0, ', ofs:', scrpStartChar:0, ' ');
39 147 --                  WritelnField(' ', @scrpFont, bStyle);
39 148 --                  WRITELN;
39 149 -2              END;
39 150 --                  newStyleSize := GetHandleSize(Handle(theTTECommand.fNewStyles));
39 151 -1              END;
39 152 --
39 153 --              WRITELN('Styles size diff: ', MAX(newStyleSize, oldStyleSize):0);
39 154 --              WRITELN('fStylePad :', theTTECommand.fStylePad:0);
39 155 -0 A    END;
39 156 --
39 157 --      {$ENDC qDebug}
39 158 --
39 159 --      {$S TERes}
39 160 --      {-----+      TSEtSelect does highlighting too, which is ugly
39 161 --      | SetSelect      |      in most places where we need to alter record
39 162 --      +-----}
39 163 -- A    PROCEDURE SetSelect(theStart, theEnd: INTEGER; hTE: TEHandle);
39 164 0- A    BEGIN
39 165 --          WITH hTE^^ DO
39 166 1-          BEGIN
39 167 --              selStart := theStart;
39 168 --              selEnd := theEnd;
39 169 -1          END;
39 170 -0 A    END;
39 171 --
39 172 --      {$S TEnonRes}
39 173 --      {-----+
39 174 --      | AutoscrollTEView      |      Callback routine from TextEdit, for auto-scrolling
39 175 --      +-----}
39 176 -- A    FUNCTION AutoscrollTEView: BOOLEAN;
39 177 --      VAR
39 178 --          msePt:    Point;
39 179 --          viewPt:   VPoint;
39 180 --          visRect:  Rect;
39 181 --          delta:    VPoint;
39 182 --          vhs:      VHSelect;
39 183 --          lead:     INTEGER;
39 184 --          trail:    INTEGER;
39 185 --          itsDown:  BOOLEAN;
39 186 --          scroller: TScroller;
39 187 --
39 188 0- A    BEGIN
39 189 --          GetMouse(msePt);
39 190 --          itsDown := StillDown;
39 191 --          IF itsDown AND (gCurrTEView <> NIL) THEN
39 192 1-          BEGIN
39 193 --              scroller := gCurrTEView.fScroller;
39 194 --              IF scroller <> NIL THEN
39 195 --                  IF (scroller.fScrollBars[h] <> NIL) OR (scroller.fScrollBars[v] <> NIL) THEN
39 196 2-                  BEGIN
39 197 --                      gCurrTEView.QDToViewPt(msePt, viewPt);

```

```

39 198 --      scroller.AutoScroll(viewPt, delta);
39 199 --      gCurrTEView.GetVisibleRect(visRect);
39 200 --
39 201 --      FOR vhs := v TO h DO
39 202 3-      BEGIN
39 203 --          lead := gCurrTEView.fLocation.vh[vhs] - visRect.topLeft.vh[vhs];
39 204 --          trail := gCurrTEView.fLocation.vh[vhs] + gCurrTEView.fSize.vh[vhs] -
39 205 --              visRect.botRight.vh[vhs];
39 206 --
39 207 --          IF delta.vh[vhs] < 0 THEN
39 208 --              delta.vh[vhs] := Min(Max(delta.vh[vhs], lead), 0)
39 209 --          ELSE
39 210 --              delta.vh[vhs] := Max(Min(delta.vh[vhs], trail), 0);
39 211 -3      END;
39 212 --      { The intent of the above is not to do autoscrolling that would scroll
39 213 --        beyond the subview boundary in any direction }
39 214 --
39 215 --      IF (delta.v <> 0) OR (delta.h <> 0) THEN
39 216 3-      BEGIN
39 217 --          scroller.ScrollBy(delta.h, delta.v, TRUE);
39 218 --
39 219 --          { includes a frame Update, which could change lots of things, thus
39 220 --            requiring us, tiresomely, to take some or all of the following
39 221 --            restorative precautions: }
39 222 --
39 223 --          IF gCurrTEView.Focus THEN
39 224 --              gCurrTEView.ClipFurtherTo(gCurrTEView.fHTE^.destRect, 0, 0);
39 225 -3      END;
39 226 -2      END;
39 227 -1      END;
39 228 --      AutoscrollTEView := TRUE;
39 229 -0 A  END;
38 624 --      {SI UTEView.TTEView.p}

```

```

40 1 --      {$P}
40 2 --      { UTEView.TTEView.p }
40 3 --      { Copyright © 1984-1988 Apple Computer Inc. All rights reserved. }
40 4 --
40 5 --      (*****
40 6 --      (*      T T E V i e w      *)
40 7 --      (*****
40 8 --
40 9 --      {$S TEOpen}
40 10 --      {-----+
40 11 --      | ITEView
40 12 --      +-----}
40 13 -- A  PROCEDURE TTEView.ITEView (itsDocument: TDocument; itsSuperView: TView;
40 14 --      itsLocation, itsSize: VPoint;
40 15 --      itsHDeterminer, itsVDeterminer: SizeDeterminer;
40 16 --      itsInset: Rect;
40 17 --      itsTextStyle: TextStyle;
40 18 --      itsJustification: INTEGER;
40 19 --      itsStyleType, itsAutoWrap: BOOLEAN);
40 20 --
40 21 0- A  BEGIN { TTEView.ITEView }
40 22 --
40 23 --      {$IFC qDebug}
40 24 --      gTEIntenseDebugging := FALSE;
40 25 --      {$ENDC}
40 26 --
40 27 --      fHTE := NIL;
40 28 --      fScroller := NIL;
40 29 --      fText := NIL;
40 30 --      fSavedTEHandle := NIL;
40 31 --      fInset := itsInset;
40 32 --      fKeyCmdNumber := cTyping;
40 33 --      fMaxChars := kUnlimited;
40 34 --      fLastHeight := 0;
40 35 --      fTypingCommand := NIL;
40 36 --      fTextStyle := itsTextStyle;
40 37 --      fJustification := itsJustification;
40 38 --      fAcceptsChanges := TRUE;
40 39 --      { Stuff to FALSE if you don't want to allow }
40 40 --      fstyleType := itsStyleType AND { _Cut, Paste, or Typing }
40 41 --      gConfiguration.hasStyleTextedit; { Only if asked, and we _can_ }
40 42 --      fAutoWrap := itsAutoWrap;
40 43 --      fFreeText := FALSE;
40 44 --      fSpecsChanged := FALSE;
40 45 --
40 46 --      fLastPageBreak := 0;
40 47 --      fLastLine := 0;
40 48 --
40 49 --      IView(itsDocument, itsSuperView, itsLocation, itsSize, itsHDeterminer, itsVDeterminer);
40 50 --
40 51 --      MakeTERecord;
40 52 --
40 53 --      defClikLoopProc := fHTE^.clikLoop; { Just in case we want to restore it... }
40 54 --      SetClikLoop(@AutoscrollTTEView, fHTE); { fHTE^.clikLoop := @TTEViewClikLoop; }
40 55 --      { ??? Things don't work well if fText is non-NIL. Is this the way to solve it? }
40 56 --      fText := fHTE^.hText;
40 57 --
40 58 --      fIdleFreq := MAX(GetCaretTime DIV 2, 1); { Makes sure I get DoIdle messages at idle }
40 59 --      { _time, so I can blink the caret }
40 60 --      IF fScroller <> NIL THEN { Set scroll parameters if in a scroller }
40 61 --      BeInScroller(fScroller);
40 62 0- A  END;
40 63 --
40 64 --
40 65 --      {$IFC qTemplateViews}
40 66 --      {$S TEOpen}
40 67 --      {-----+
40 68 --      | IRes
40 69 --      +-----}
40 70 -- A  PROCEDURE TTEView.IRes (itsDocument: TDocument; itsSuperView: TView; VAR itsParams: Ptr); OVERRIDE;
40 71 --
40 72 --      VAR
40 73 --      aTextStyle: TextStyle;
40 74 --
40 75 0- A  BEGIN
40 76 --      {$IFC qDebug}
40 77 --      gTEIntenseDebugging := FALSE;
40 78 --      {$ENDC}
40 79 --
40 80 --      fHTE := NIL; { In case of emergency. }
40 81 --      fScroller := NIL;
40 82 --      fText := NIL;
40 83 --      fSavedTEHandle := NIL;
40 84 --      fLastPageBreak := 0;
40 85 --      fLastLine := 0;
40 86 --
40 87 --      INHERITED IRes(itsDocument, itsSuperView, itsParams);
40 88 --
40 89 --      WITH TTEViewTemplatePtr(itsParams)^ DO
40 90 1-  BEGIN
40 91 --      fTypingCommand := NIL;
40 92 --      fLastHeight := 0;
40 93 --      fSpecsChanged := FALSE;
40 94 --
40 95 --      fInset := itsInset;
40 96 --      fKeyCmdNumber := itsKeyCmdNumber;
40 97 --      fMaxChars := itsMaxChars;
40 98 --      SetTextStyle(aTextStyle,

```

```

40 99 --      GetFontNum(itsFontName), itsTextFace, itsTextSize, itsTextColor);
40 100 --      fTextStyle := aTextStyle;
40 101 --      fJustification := itsJustification;
40 102 --      fAcceptsChanges := itsAcceptsChanges;
40 103 --      fStyleType := itsStyleType AND gConfiguration.hasStyleTextedit;
40 104 --      fAutoWrap := itsAutoWrap;
40 105 --      fFreeText := itsFreesText;
40 106 --
40 107 --      MakeTERecord;
40 108 --
40 109 --      defClickLoopProc := fhTE^.clickLoop;  ( Just in case we want to restore it_ )
40 110 --      SetClickLoop(@AutoscrollTEView, fhTE);  ( fhTE^.clickLoop := @TEViewClickLoop; )
40 111 --      fText := fhTE^.hText;
40 112 --
40 113 --      fIdleFreq := MAX(GetCaretTime DIV 2,1);  ( Makes sure I get DoIdle messages at idle )
40 114 --      ( _time, so I can blink the caret )
40 115 --1      END;
40 116 --
40 117 --      OffsetPtrWStr(itsParams, SIZEOF(TEViewTemplate));
40 118 --0 A  END;
40 119 --      { $ENDC }
40 120 --
40 121 --
40 122 --      { $IFC qWriteTemplates }
40 123 --      { $S MAWriteRes }
40 124 --      {-----+
40 125 --      | WRes
40 126 --      +-----+ }
40 127 -- A  PROCEDURE TTEView.WRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
40 128 --  VAR
40 129 --      theFont: Str255;
40 130 --      ttPtr: TEViewTemplatePtr;
40 131 --0 A  BEGIN
40 132 --      INHERITED WRes(theResource, itsParams);
40 133 --
40 134 --      GetFontName(fTextStyle.tsFont, theFont);
40 135 --
40 136 --      ttPtr := TEViewTemplatePtr(ExpandPtrWStr(theResource, itsparams,
40 137 --      SIZEOF(TEViewTemplate), LENGTH(theFont)));
40 138 --
40 139 --      WITH ttPtr^ DO
40 140 --1      BEGIN
40 141 --      itsStyleType := fStyleType;
40 142 --      itsAutoWrap := fAutoWrap;
40 143 --      itsAcceptsChanges := fAcceptsChanges;
40 144 --      itsFreesText := fFreeText;
40 145 --      itsKeyCmdNumber := fKeyCmdNumber;
40 146 --      itsMaxChars := fMaxChars;
40 147 --      itsInset := fInset;
40 148 --      itsJustification := fJustification;
40 149 --      WITH fTextStyle DO
40 150 --2      BEGIN
40 151 --      itsTextFace := tsFace;
40 152 --      itsTextSize := tsSize;
40 153 --      itsTextColor := tsColor;
40 154 --2      END;
40 155 --      (* itsFontName := theFont; *)
40 156 --      CopyStr255(theFont, PRStr(itsFontName));
40 157 --1      END;
40 158 --0 A  END;
40 159 --
40 160 --      { $S MAWriteRes }
40 161 --      {-----+
40 162 --      | WriteRes
40 163 --      +-----+ }
40 164 -- A  PROCEDURE TTEView.WriteRes (theResource: ViewRsrcHndl; VAR itsParams: Ptr); OVERRIDE;
40 165 --0 A  BEGIN
40 166 --      gWResSignature := 'tevw'; gWResType := 'TTEView';
40 167 --      WRes(theResource, itsParams);
40 168 --0 A  END;
40 169 --      { $ENDC }
40 170 --
40 171 --
40 172 --      { $S TEClose }
40 173 --      {-----+
40 174 --      | Free
40 175 --      +-----+ }
40 176 -- A  PROCEDURE TTEView.Free; OVERRIDE;
40 177 --0 A  BEGIN
40 178 --      IF fhTE <> NIL THEN
40 179 --1      BEGIN
40 180 --      IF fSavedTEHandle <> NIL THEN
40 181 --          WITH fhTE^ DO
40 182 --2      BEGIN
40 183 --          hText := fSavedTEHandle;  ( Worry about fText separately. Put back )
40 184 --          ($H-)  ( _the handle which TE allocated )
40 185 --          teLength :=  ( This is here because it only makes )
40 186 --          GetHandleSize(hText);
40 187 --          IF fFreeText THEN  ( _sense if fSavedTEHandle is not NIL. )
40 188 --              DisposalHandle(fText);
40 189 --          ($H+)
40 190 --2      END;
40 191 --          TEDispose(fhTE);
40 192 --          fSavedTEHandle := NIL;
40 193 --1      END;
40 194 --      INHERITED Free;
40 195 --0 A  END;
40 196 --
40 197 --      { $S TENonRes }

```



```

40 198 -- {-----+
40 199 -- |   AutoScrolling   |
40 200 -- +-----+
40 201 -- A  PROCEDURE TTEView.AutoScrolling (doScrolling: BOOLEAN);
40 202 -- 0- A  BEGIN
40 203 --      ($IFC NOT qNeedsROM128K)
40 204 --      IF gConfiguration.hasROM128K THEN
40 205 --      ($ENDC)
40 206 --          IF fhTE <> NIL THEN
40 207 --              TEAutoView(doScrolling, fhTE);
40 208 -- 0 A  END;
40 209 --
40 210 --      {$S TENonRes}
40 211 --      {-----+
40 212 --      |   BeInPort   |
40 213 --      +-----+
40 214 -- A  PROCEDURE TTEView.BeInPort(itsPort: GrafPtr); OVERRIDE;
40 215 -- 0- A  BEGIN
40 216 --
40 217 --      IF fhTE <> NIL THEN
40 218 --      BEGIN
40 219 --          WITH fhTE^^ DO
40 220 --              IF itsPort = NIL THEN
40 221 --                  inPort := gWorkPort
40 222 --              ELSE
40 223 --                  inPort := itsPort;
40 224 --
40 225 --              IF itsPort = NIL THEN
40 226 --      BEGIN
40 227 --          DoneTyping;
40 228 --          fSpecsChanged := TRUE;
40 229 --      END;
40 230 --      END;
40 231 --
40 232 --      INHERITED BeInPort(itsPort);
40 233 -- 0 A  END;
40 234 --
40 235 --
40 236 --      {$S TENonRes}
40 237 --      {-----+
40 238 --      |   BeInScroller   |
40 239 --      +-----+
40 240 -- A  PROCEDURE TTEView.BeInScroller(itsScroller: TScroller); OVERRIDE;
40 241 -- VAR
40 242 --     vertScrollUnit:    INTEGER;
40 243 -- 0- A  BEGIN
40 244 --     fScroller := itsScroller;
40 245 --     IF (fhTE <> NIL) & (itsScroller <> NIL) THEN
40 246 --     BEGIN
40 247 --         IF fhTE^^.lineHeight > 0 THEN      { This works for both old & new TextEdit }
40 248 --             vertScrollUnit := fhTE^^.lineHeight
40 249 --         ELSE IF gConfiguration.hasScriptManager THEN { Ask for system default size }
40 250 --             vertScrollUnit := GetDefFontSize
40 251 --         ELSE
40 252 --             vertScrollUnit := kStdScrollUnit;
40 253 --
40 254 --             itsScroller.SetScrollParameters(kStdScrollUnit, vertScrollUnit, FALSE, TRUE);
40 255 --     END;
40 256 --
40 257 --     INHERITED BeInScroller(itsScroller);
40 258 -- 0 A  END;
40 259 --
40 260 --
40 261 --      {$S TENonRes}
40 262 --      {-----+
40 263 --      |   CalcMinSize   |
40 264 --      +-----+
40 265 -- A  PROCEDURE TTEView.CalcMinSize(VAR minSize: VPoint); OVERRIDE;
40 266 -- VAR
40 267 --     selLoc:      Point;
40 268 --     theHeight:   INTEGER;
40 269 --     theAscent:   INTEGER;
40 270 --     vHite:       LONGINT;
40 271 --     vhs:         VHSelect;
40 272 --     theLines:    INTEGER;
40 273 --     theLength:   INTEGER;
40 274 --     theStyle:    TextStyle;
40 275 --
40 276 -- 0- A  BEGIN
40 277 --     vHite := CalcRealHeight;
40 278 --
40 279 --     { Note that we omit the margins here, so that if TView.ComputeExtent rounds up to a
40 280 --       page multiple, the margins will get tacked on after.  Also, insure we don't run off
40 281 --       the end of the coordinate system if there are many lines of text. }
40 282 --
40 283 --     SetVPt(minSize, fSize.h - fInset.left - fInset.right, MIN(kMaxCoord, vHite));
40 284 --
40 285 --     FOR vhs := v TO h DO
40 286 --         IF fSizeDeterminer[vhs] = sizeFixed THEN
40 287 --             minSize.vh[vhs] := fSize.vh[vhs];
40 288 -- 0 A  END;
40 289 --
40 290 --
40 291 --      {$S TENonRes}
40 292 --      {-----+
40 293 --      |   CalcRealHeight   |
40 294 --      +-----+
40 295 -- A  FUNCTION TTEView.CalcRealHeight: INTEGER;
40 296 -- VAR

```

```

40 297 --      lastIsCR:      BOOLEAN;
40 298 --      theMode:      INTEGER;
40 299 --      theHeight:    INTEGER;
40 300 --      theStyle:     TextStyle;
40 301 --      theFontInfo:  FontInfo;
40 302 --      savedPort:    GrafPtr;
40 303 0- A BEGIN
40 304 --      ($IFC qRangeCheck){$R-}($ENDC)
40 305 --      WITH fHTE^^ DO
40 306 --          lastIsCR := (teLength <= 0) |
40 307 --                      (CharsHandle(hText)^^[PRED(teLength)] = chReturn);
40 308 --      ($IFC qRangeCheck){$R+}($ENDC)
40 309 --
40 310 --      IF fStyleType = kWithStyle THEN
40 311 1- BEGIN
40 312 --          theHeight := 0;
40 313 --          IF fHTE^^.nLines > 0 THEN
40 314 --              theHeight := TEGetHeight(0, MAXINT, fHTE);
40 315 --
40 316 --          IF lastIsCR THEN                                { then can't use TEGetHeight so we }
40 317 2- BEGIN                                                { _have to figure it out ourselves. }
40 318 --              theMode := doAll;
40 319 --              lastIsCR := ContinuousStyle(MAXINT, MAXINT, theMode, theStyle);
40 320 --              GetPort(savedPort);
40 321 --              SetPort(gWorkPort);
40 322 --              SetPortTextStyle(theStyle);
40 323 --              GetFontInfo(theFontInfo);                    { What a pain... }
40 324 --              SetPort(savedPort);
40 325 --              WITH theFontInfo DO
40 326 --                  theHeight := theHeight + ascent + descent + leading;
40 327 2- END;
40 328 1- END
40 329 --      ELSE
40 330 1- BEGIN
40 331 --          theHeight := fHTE^^.nLines + ORD(lastIsCR);
40 332 --          theHeight := theHeight * fHTE^^.lineHeight;
40 333 1- END;
40 334 --
40 335 --      CalcRealHeight := theHeight;
40 336 --      ($IFC qDebug)
40 337 --      IF gIntenseDebugging THEN
40 338 --          WRITELN('CalcRealHeight=', theHeight:0);
40 339 --      ($ENDC)
40 340 0- A END;
40 341 --
40 342 --
40 343 --      {$S TNonRes}
40 344 --      {-----+
40 345 --      | ChangeWrap |
40 346 --      +-----}
40 347 0- A PROCEDURE TTEView.ChangeWrap (newAutoWrap, redraw: BOOLEAN);
40 348 0- A BEGIN
40 349 --          fAutoWrap := newAutoWrap;
40 350 --          WITH fHTE^^ DO
40 351 --              IF newAutoWrap THEN
40 352 --                  crOnly := 0
40 353 --              ELSE
40 354 --                  crOnly := -1;
40 355 --              IF redraw THEN
40 356 1- BEGIN
40 357 --                  RecalcText;
40 358 --                  SynchView(kRedraw);
40 359 1- END;
40 360 0- A END;
40 361 --
40 362 --      {$S TNonRes}
40 363 --      {-----+
40 364 --      | ComputeSize |
40 365 --      +-----}
40 366 0- A PROCEDURE TTEView.ComputeSize(VAR newSize: VPoint); OVERRIDE;
40 367 --
40 368 --      {-----+
40 369 --      | NeedAdjust |
40 370 --      +-----}
40 371 0- B FUNCTION NeedAdjust(vhs: VHSelect): BOOLEAN;
40 372 0- B BEGIN
40 373 --      NeedAdjust := NOT (fSizeDeterminer[vhs] IN
40 374 --                          [sizeFixed, sizeSuperView, sizeRelSuperView]);
40 375 0- B END;
40 376 --
40 377 0- A BEGIN { TTEView.ComputeSize }
40 378 --      INHERITED ComputeSize(newSize);
40 379 --
40 380 --      IF NeedAdjust(h) THEN                                { If necessary, tack on the margins }
40 381 --          newSize.h := MIN(kMaxCoord, ORD4(newSize.h) + fInset.left + fInset.right);
40 382 --      IF NeedAdjust(v) THEN
40 383 --          newSize.v := MIN(kMaxCoord, ORD4(newSize.v) + fInset.top + fInset.bottom);
40 384 0- A END;
40 385 --
40 386 --
40 387 --      {$S TRes}
40 388 --      {-----+ Assumes grafPort set appropriately upon entry, so
40 389 --      | CalcSelLoc | that TextWidth gives width in screen resolution
40 390 --      +-----}
40 391 0- A PROCEDURE TTEView.CalcSelLoc(VAR selectionRect: Rect);
40 392 --
40 393 --      CONST
40 394 --          teSysJust = $BAC;                                { System justification direction }
40 395 --          kSlopToAllow = 36;

```

```

40 396 --
40 397 --   VAR
40 398 --       handleToText:   CharsHandle;
40 399 --       startOfSelection: INTEGER;
40 400 --       endOfSelection:   INTEGER;
40 401 --       startOfCurrentLine: INTEGER;
40 402 --       startOfNextLine:  INTEGER;
40 403 --       charCount:        INTEGER;
40 404 --       selLine:           INTEGER;
40 405 --       justification:     INTEGER;
40 406 --       upToSelWidth:      INTEGER;
40 407 --       restWidth:         INTEGER;
40 408 --       lineHeight:        INTEGER;
40 409 --       fontAscent:         INTEGER;
40 410 --       theStyle:           TextStyle;
40 411 --       keyDirection:       SignedByte;
40 412 --       lastIsReturn:       BOOLEAN;
40 413 --       theMode:            INTEGER;
40 414 --       dontCare:           BOOLEAN;
40 415 --       savedPort:          GrafPtr;
40 416 --       theFontInfo:        FontInfo;
40 417 --
40 418 0- A BEGIN
40 419 --   WITH fhTE^^ DO
40 420 1-   BEGIN
40 421 --       charCount := teLength;
40 422 --       startOfSelection := selStart;
40 423 --       endOfSelection := selEnd;
40 424 --       handleToText := CharsHandle(hText);
40 425 --       justification := just;
40 426 -1   END;
40 427 --
40 428 --   {$IFC qRangeCheck}{$R-}{$ENDC}
40 429 --   lastIsReturn := (startOfSelection = charCount) & (charCount > 0) &
40 430 --                   (handleToText^[charCount - 1] = chReturn);
40 431 --   {$IFC qRangeCheck}{$R+}{$ENDC}
40 432 --
40 433 --   IF gConfiguration.hasStyleTextEdit THEN
40 434 1-   BEGIN
40 435 --       selectionRect.topLeft := TEGetPoint(startOfSelection, fhTE);
40 436 --       TEGetStyle(startOfSelection, theStyle, lineHeight, fontAscent, fhTE);
40 437 --       selectionRect.top := selectionRect.top - lineHeight;
40 438 --       selectionRect.botRight := TEGetPoint(endOfSelection, fhTE);
40 439 --
40 440 --       IF lastIsReturn THEN
40 441 2-   BEGIN
40 442 --           theMode := doAll;
40 443 --           dontCare := ContinuousStyle(MAXINT, MAXINT, theMode, theStyle);
40 444 --           GetPort(savedPort);
40 445 --           SetPort(gWorkPort);
40 446 --           SetPortTextStyle(theStyle);
40 447 --           GetFontInfo(theFontInfo); { What a pain... }
40 448 --           SetPort(savedPort);
40 449 --           selectionRect.top := selectionRect.bottom;
40 450 --           WITH theFontInfo DO
40 451 --               selectionRect.bottom := selectionRect.top + ascent + descent + leading;
40 452 -2   END;
40 453 -1   END
40 454 --   ELSE
40 455 1-   BEGIN
40 456 --
40 457 --       WITH fhTE^^ DO { for lineStarts and nLines }
40 458 2-   BEGIN
40 459 --           selLine := 1;
40 460 --           WHILE (selLine < nLines) &
40 461 --                 (startOfSelection >= lineStarts[selLine]) DO
40 462 --               selLine := selLine + 1;
40 463 --           selLine := selLine - 1;
40 464 --
40 465 --           IF lastIsReturn |
40 466 --               (justification <> teJustLeft) AND (startOfSelection = lineStarts[SUCC(selLine)]) THEN
40 467 --               selLine := SUCC(selLine);
40 468 --
40 469 --           selectionRect.top := (lineHeight * selLine) + finset.top;
40 470 --           startOfCurrentLine := lineStarts[selLine];
40 471 --           IF selLine < nLines THEN
40 472 --               startOfNextLine := lineStarts[SUCC(selLine)]
40 473 --           ELSE
40 474 --               startOfNextLine := charCount;
40 475 -2   END;
40 476 --
40 477 --   { We could move the handle high first, but this routine is called frequently and
40 478 --     fragmentation shouldn't be a problem for the TextWidth call--the worst that will
40 479 --     happen is the Font Manager will have to substitute a font & we'll be off slightly. }
40 480 --   { ??? The following Text calls may be unnecessary, since this may only get done
40 481 --     after TextEdit has run and set the font already. This appears not to be the case
40 482 --     for the Select All command, however, and is not true in general. }
40 483 --   TextFont(fhTE^^.txFont);
40 484 --   TextFace(fhTE^^.txFace);
40 485 --   TextSize(fhTE^^.txSize);
40 486 --   HLock(Handle(handleToText));
40 487 --
40 488 --   { Figure out where on the line the selection is. If the Script Manager is
40 489 --     installed, use it. Otherwise, try and figure it out ourselves. }
40 490 --   IF gConfiguration.hasScriptManager THEN
40 491 2-   BEGIN
40 492 --       { Unfortunately, we must get this every time, as the user can change script
40 493 --         directions at a moment's notice. }
40 494 --       IF startOfSelection = endOfSelection THEN

```

```

40 495 --      keyDirection := GetScript(GetEnvirons(smKeyScript), smScriptRight)
40 496 --      ELSE
40 497 --      keyDirection := smHilite;
40 498 --
40 499 --      upToSelWidth := Char2Pixel(Ptr(ORD(handleToText^)+startOfCurrentLine),
40 500 --      startOfNextLine-startOfCurrentLine, 0,
40 501 --      startOfSelection-startOfCurrentLine,
40 502 --      keyDirection);
40 503 --      restWidth := TextWidth(QDPtr(handleToText^), startOfCurrentLine,
40 504 --      startOfNextLine - startOfCurrentLine) - upToSelWidth;
40 505 --      END
40 506 --      ELSE
40 507 --      BEGIN
40 508 --      upToSelWidth := TextWidth(QDPtr(handleToText^), startOfCurrentLine,
40 509 --      startOfSelection - startOfCurrentLine);
40 510 --      restWidth := TextWidth(QDPtr(handleToText^), startOfSelection,
40 511 --      startOfNextLine - startOfSelection);
40 512 --      END;
40 513 --      HUnlock(Handle(handleToText));
40 514 --      CASE justification OF
40 515 --      teJustLeft:
40 516 --      selectionRect.left := fInset.left + upToSelWidth;
40 517 --      teJustRight:
40 518 --      selectionRect.left := fSize.h - fInset.right - restWidth;
40 519 --      { Following looks somewhat baroque, but avoids integer overflow }
40 520 --      teJustCenter:
40 521 --      selectionRect.left := fSize.h DIV 2 - restWidth DIV 2 + upToSelWidth DIV 2;
40 522 --      END;
40 523 --
40 524 --      WITH selectionRect DO
40 525 --      BEGIN
40 526 --      left := MAX(left - 2, 0);
40 527 --      IF left <= kSlopToAllow THEN
40 528 --      left := 0;
40 529 --      { show start of line if close; gets around }
40 530 --      { _potential annoyance assoc w/typeahead }
40 531 --      ($IFC FALSE)
40 532 --      top := MAX(top - 2, 0);
40 533 --      ($ENDC)
40 534 --      right := MIN(left + 8, fSize.h);
40 535 --      bottom := MIN(top + fHTE^.lineHeight, fSize.v);
40 536 --      END;
40 537 --      END;
40 538 --
40 539 --      ($$ TEnonRes)
40 540 --      {-----+
40 541 --      | ContinuousStyle
40 542 --      +-----}
40 543 --      A FUNCTION TTEView.ContinuousStyle(firstChar, lastChar: INTEGER; VAR mode: INTEGER;
40 544 --      VAR aStyle: TextStyle): BOOLEAN;
40 545 --      VAR
40 546 --      oldSelStart,
40 547 --      oldSelEnd: INTEGER;
40 548 --      A BEGIN
40 549 --      WITH fHTE^^ DO
40 550 --      BEGIN
40 551 --      oldSelStart := selStart;
40 552 --      oldSelEnd := selEnd;
40 553 --      END;
40 554 --      SetSelect(firstChar, lastChar, fHTE); { Use SetSelect so this is invisible }
40 555 --      ContinuousStyle := TEContinuousStyle(mode, aStyle, fHTE);
40 556 --      SetSelect(oldSelStart, oldSelEnd, fHTE);
40 557 --      A END;
40 558 --
40 559 --      ($$ TERes)
40 560 --      {-----+
40 561 --      | ContainsClipType
40 562 --      +-----}
40 563 --      A FUNCTION TTEView.ContainsClipType(aType: ResType): BOOLEAN; OVERRIDE;
40 564 --      A BEGIN
40 565 --      ContainsClipType := (aType = 'TEXT');
40 566 --      A END;
40 567 --
40 568 --      { Put in resident segment since this gets called while typing }
40 569 --      ($$ TERes)
40 570 --      {-----+
40 571 --      | DoBreakFollowing
40 572 --      +-----}
40 573 --      A FUNCTION TTEView.DoBreakFollowing(vhs: VHSelect; prevBreak: VCoordinate;
40 574 --      VAR automatic: BOOLEAN): VCoordinate; OVERRIDE;
40 575 --      VAR
40 576 --      orthoVhs: VHSelect;
40 577 --      possibleLoc: INTEGER;
40 578 --      theStyles: TStyleHandle;
40 579 --      lhTab: LHHandle;
40 580 --      height,
40 581 --      lineHeight: INTEGER;
40 582 --      lineNo: INTEGER;
40 583 --
40 584 --      A BEGIN
40 585 --      orthoVhs := gOrthogonal[vhs];
40 586 --      automatic := TRUE;
40 587 --
40 588 --      possibleLoc := MIN(kMaxCoord, ORD4(prevBreak) + fPrintHandler.fViewPerPage.vh[orthoVhs]);
40 589 --
40 590 --      { We want to get rid of the on-screen margin represented by fMargin when printing, so
40 591 --      adjust things so that the portion of the view occupied by the screen margin doesn't
40 592 --      get printed. }
40 593 --      IF prevBreak = 0 THEN

```

```

40 594 --           possibleLoc := possibleLoc + fInset.topLeft.vh[orthoVhs];
40 595 --
40 596 --           IF (fStyleType = kWithStyle) AND (vhs = h) THEN
40 597 1-             BEGIN
40 598 --             IF (fLastPageBreak = prevBreak) THEN
40 599 2-               BEGIN
40 600 --                 height := fLastPageBreak;
40 601 --                 lineNo := fLastLine;
40 602 -2              END
40 603 --             ELSE
40 604 2-               BEGIN
40 605 --                 height := fInset.topLeft.vh[orthoVhs];
40 606 --                 lineNo := 0;
40 607 -2              END;
40 608 --
40 609 --             theStyles := GetStylHandle(fHTE);
40 610 --             lhTab := theStyles^^.lhTab;
40 611 --             WHILE lineNo < fHTE^^.nLines DO
40 612 2-               BEGIN
40 613 --                 lineHeight := lhTab^^[lineNo].lhHeight;
40 614 --                 IF height + lineHeight <= possibleLoc THEN
40 615 --                   height := height + lineHeight
40 616 --                 ELSE
40 617 --                   LEAVE;
40 618 --                   lineNo := lineNo + 1;
40 619 -2              END;
40 620 --             IF lineNo >= fHTE^^.nLines THEN
40 621 --               possibleLoc := Max(possibleLoc, height)
40 622 --             ELSE
40 623 --               possibleLoc := height;
40 624 --               fLastPageBreak := possibleLoc;
40 625 --               fLastLine := lineNo;
40 626 -1             END;
40 627 --
40 628 --
40 629 --           IF ORD4(possibleLoc + fInset.topLeft.vh[orthoVhs])
40 630 --             >= fSize.vh[orthoVhs] THEN
40 631 --             DoBreakFollowing := fSize.vh[orthoVhs]
40 632 --           ELSE
40 633 --             DoBreakFollowing := possibleLoc;
40 634 -0 A         END;
40 635 --
40 636 --
40 637 --           {$S TENonRes}
40 638 --           {-----+
40 639 --           | DoCalcViewPerPage |
40 640 --           +-----}
40 641 -- A         PROCEDURE TTEView.DoCalcViewPerPage (VAR viewPerPage: VPoint); OVERRIDE;
40 642 --         VAR
40 643 --           vhs:           VHSelect;
40 644 --
40 645 0- A       BEGIN
40 646 --           INHERITED DoCalcViewPerPage(viewPerPage); { Get max amount allowed given margins }
40 647 --
40 648 --           {$IFC qDebug}
40 649 --           IF gDebugPrinting THEN
40 650 1-             BEGIN
40 651 --               WRITE('TTEView: incoming generic viewPerPage:');
40 652 --               WriteVPt(viewPerPage);
40 653 --               WRITELN;
40 654 -1            END;
40 655 --           {$ENDC}
40 656 --
40 657 --           IF (fStyleType <> kWithStyle) AND (fHTE <> NIL) THEN { Adjust for integral # of lines per page }
40 658 --             WITH fHTE^^ DO
40 659 --               viewPerPage.v := lineHeight * (viewPerPage.v DIV lineHeight);
40 660 --
40 661 --           {$IFC qDebug}
40 662 --           IF gDebugPrinting THEN
40 663 1-             BEGIN
40 664 --               WRITE('TTEView: computed viewPerPage:');
40 665 --               WriteVPt(viewPerPage);
40 666 --               WRITELN;
40 667 -1            END;
40 668 --           {$ENDC}
40 669 -0 A       END;
40 670 --
40 671 --           {$S TERes}
40 672 --           {-----+
40 673 --           | DoIdle |
40 674 --           +-----}
40 675 -- A         PROCEDURE TTEView.DoIdle(phase: IdlePhase); OVERRIDE;
40 676 --         VAR
40 677 --           aRect: Rect;
40 678 --
40 679 0- A       BEGIN
40 680 --           IF (fHTE <> NIL) & fAcceptsChanges & Focus THEN
40 681 1-             BEGIN
40 682 --               TEIdle(fHTE);
40 683 --               fIdleFreq := MAX(GetCaretTime DIV 2, 1); { Reset idle frequency in case user }
40 684 -1            END; { _changed cursor blink time }
40 685 -0 A       END;
40 686 --
40 687 --
40 688 --           {$S TERes}
40 689 --           { The Tab character has no width in some fonts, and so can cause confusing screen feedback.
40 690 --             If you want to filter it out, OVERRIDE DoKeyCommand and either translate it to something
40 691 --             else, or throw it away (remembering to return gNoChanges). }
40 692 --           {-----+

```

```

40 693 -- | DoKeyCommand |
40 694 -- +-----+
40 695 -- A FUNCTION TTEView.DoKeyCommand(ch: Char; aKeyCode: INTEGER; VAR info: EventInfo): TCommand; OVERRIDE;
40 696 -- VAR
40 697 --     aTypingCommand: TTETypingCommand;
40 698 --     needNewCommand: BOOLEAN;
40 699 --
40 700 0- A BEGIN
40 701 --     DoKeyCommand := gNoChanges;
40 702 --     IF fAcceptsChanges THEN
40 703 1- BEGIN
40 704 --         IF Focus THEN ; { Try to focus before accepting the key. }
40 705 --         IF (ch >= chLeft) AND { If the user typed a cursor key, handle it }
40 706 --             (ch <= chDown) THEN { _like a mouse down. }
40 707 2- BEGIN
40 708 --             DoneTyping; { Like mousedown, further typing = new cmd }
40 709 --             fSpecsChanged := TRUE;
40 710 --             TEKey(ch, fHTE);
40 711 --             ScrollSelectionIntoView;
40 712 --             EXIT(DoKeyCommand);
40 713 -2 END;
40 714 --
40 715 --     IF ch = chClear THEN { Hitting the Clear key is equivalent to }
40 716 2- BEGIN { _choosing Clear from the Edit menu. }
40 717 --         {$H-}
40 718 --         WITH fHTE^^ DO { If nothing is selected, does nothing. }
40 719 --             IF selStart < selEnd THEN
40 720 --                 DoKeyCommand := DoMakeEditCommand(cClear);
40 721 --             EXIT(DoKeyCommand);
40 722 --             {$H+}
40 723 -2 END;
40 724 --
40 725 --     IF ch = chEnter THEN { If we get an Enter, throw it away }
40 726 --         EXIT(DoKeyCommand);
40 727 --
40 728 --     IF (ch <> chBackspace) AND { Check max size for text, and that we're }
40 729 --         (fHTE^^.selStart = fHTE^^.selEnd) THEN { _not running out of memory }
40 730 --         IF ((fMaxChars - GetHandleSize(fText)) < 1) OR MemSpaceIsLow THEN
40 731 2- BEGIN
40 732 --             SysBeep(0);
40 733 --             EXIT(DoKeyCommand); { Flush further keystrokes }
40 734 -2 END;
40 735 --
40 736 --     needNewCommand := (fTypingCommand = NIL);
40 737 --     IF NOT needNewCommand THEN
40 738 --         needNewCommand := fTypingCommand.fCompleted;
40 739 --
40 740 --     IF needNewCommand THEN
40 741 2- BEGIN
40 742 --         aTypingCommand := DoMakeTypingCommand(ch);
40 743 --         fTypingCommand := aTypingCommand;
40 744 --         DoKeyCommand := aTypingCommand;
40 745 -2 END
40 746 --     ELSE
40 747 --         fTypingCommand.AddCharacter(ch);
40 748 --
40 749 -1 END;
40 750 0- A END;
40 751 --
40 752 --
40 753 -- {$S TEselCommand}
40 754 -- {-----+
40 755 -- | DoMakeEditCommand |
40 756 -- +-----}
40 757 -- A FUNCTION TTEView.DoMakeEditCommand(aCmdNumber: CmdNumber): TTECommand;
40 758 -- VAR
40 759 --     aCutCopyCommand: TTECutCopyCommand;
40 760 --     aPasteCommand: TTEPasteCommand;
40 761 --     aClearCommand: TTECommand;
40 762 --
40 763 0- A BEGIN
40 764 1- CASE aCmdNumber OF
40 765 --     cCut,
40 766 2- cCopy: BEGIN
40 767 --         New(aCutCopyCommand);
40 768 --         FailNIL(aCutCopyCommand);
40 769 --         aCutCopyCommand.ITECutCopyCommand(SELF, aCmdNumber);
40 770 --         DoMakeEditCommand := aCutCopyCommand;
40 771 -2 END;
40 772 --
40 773 2- cPaste: BEGIN
40 774 --         New(aPasteCommand);
40 775 --         FailNIL(aPasteCommand);
40 776 --         aPasteCommand.ITEPasteCommand(SELF);
40 777 --         DoMakeEditCommand := aPasteCommand;
40 778 -2 END;
40 779 --
40 780 2- cClear: BEGIN
40 781 --         New(aClearCommand);
40 782 --         FailNIL(aClearCommand);
40 783 --         aClearCommand.ITEClearCommand(SELF, aCmdNumber, TRUE);
40 784 --         DoMakeEditCommand := aClearCommand;
40 785 -2 END;
40 786 --
40 787 -1 END;
40 788 0- A END;
40 789 --
40 790 --
40 791 -- {$S TERes}

```

```

40 792 -- {-----+
40 793 -- | DoMakeStyleCommand |
40 794 -- +-----+
40 795 -- A FUNCTION TTEView.DoMakeStyleCommand(aStyle: TextStyle; itsCmdNumber: CmdNumber;
40 796 --                                     itsMode: INTEGER): TTEStyleCommand;
40 797 -- VAR
40 798 --     aTEStyleCommand: TTEStyleCommand;
40 799 --
40 800 -- A BEGIN
40 801 --     New(aTEStyleCommand);
40 802 --     FailNil(aTEStyleCommand);
40 803 --     aTEStyleCommand.ITTEStyleCommand(SELF, aStyle, itsCmdNumber, itsMode);
40 804 --     DoMakeStyleCommand := aTEStyleCommand;
40 805 -- A END;
40 806 --
40 807 --
40 808 -- {$$ TRes}
40 809 -- {-----+
40 810 -- | DoMakeTypingCommand |
40 811 -- +-----+
40 812 -- A FUNCTION TTEView.DoMakeTypingCommand(ch: Char): TTETypingCommand;
40 813 -- VAR
40 814 --     aTypingCommand: TTETypingCommand;
40 815 --
40 816 -- A BEGIN
40 817 --     New(aTypingCommand);
40 818 --     FailNil(aTypingCommand);
40 819 --     aTypingCommand.ITETypingCommand(SELF, ch);
40 820 --     DoMakeTypingCommand := aTypingCommand;
40 821 -- A END;
40 822 --
40 823 --
40 824 -- {$$ TSElCommand}
40 825 -- {-----+
40 826 -- | DoMenuCommand |
40 827 -- +-----+
40 828 -- A FUNCTION TTEView.DoMenuCommand(aCmdNumber: CmdNumber): TCommand; OVERRIDE;
40 829 -- VAR
40 830 --     command: TCommand;
40 831 --     aCutCopyCommand: TTECutCopyCommand;
40 832 --     aPasteCommand: TTEPasteCommand;
40 833 --     nChars: LONGINT;
40 834 --     dataType: ResType;
40 835 --
40 836 -- A BEGIN
40 837 --     DoMenuCommand := gNoChanges;
40 838 --     CASE aCmdNumber OF
40 839 --         cCut,
40 840 --         cCopy,
40 841 --         cClear:
40 842 --             DoMenuCommand := DoMakeEditCommand(aCmdNumber);
40 843 --
40 844 --         cPaste:
40 845 --             BEGIN
40 846 --                 nChars := gApplication.GetDataToPaste(NIL, dataType);
40 847 --                 IF nChars < 0 THEN
40 848 --                     ($IFC qDebug) ProgramBreak('Couldn't get data to paste') { ??? }
40 849 --                 ($ENDC)
40 850 --             ELSE
40 851 --                 BEGIN
40 852 --                     IF nChars - (fHTE^.selEnd - fHTE^.selStart) >
40 853 --                         fMaxChars - GetHandleSize(fText) THEN
40 854 --                             StdAlert(phTooManyChars)
40 855 --                         ELSE
40 856 --                             DoMenuCommand := DoMakeEditCommand(aCmdNumber);
40 857 --                         END;
40 858 --                     END;
40 859 --                 END;
40 860 --             OTHERWISE
40 861 --                 cSelectAll:
40 862 --                     BEGIN
40 863 --                         IF Focus THEN
40 864 --                             BEGIN
40 865 --                                 TSElSelect(Q, fHTE^.teLength, fHTE);
40 866 --                                 DoneTyping;
40 867 --                                 fSpecsChanged := TRUE;
40 868 --                                 ScrollSelectionIntoView;
40 869 --                             END;
40 870 --                         OTHERWISE
40 871 --                             DoMenuCommand := INHERITED DoMenuCommand(aCmdNumber);
40 872 --                         END;
40 873 --                     END;
40 874 --                 OTHERWISE
40 875 --                     END;
40 876 --             END;
40 877 --         END;
40 878 --
40 879 -- {$$ TRes}
40 880 -- {-----+
40 881 -- | DoMouseCommand |
40 882 -- +-----+
40 883 -- A FUNCTION TTEView.DoMouseCommand(VAR theMouse: Point; VAR info: EventInfo;
40 884 --                                     VAR hysteresis: Point): TCommand; OVERRIDE;
40 885 -- A BEGIN
40 886 --     IF Focus THEN
40 887 --         BEGIN
40 888 --             gCurrTEView := SELF;
40 889 --             DoneTyping;
40 890 --             fSpecsChanged := TRUE;

```

{ Mousedown terminates the Typing command }

```

40 891 --      TClick(theMouse, info.theShiftKey, fHTE);
40 892 -1      END;
40 893 --      DoMouseCommand := gNoChanges;
40 894 -0 A  END;
40 895 --
40 896 --
40 897 --      {$S TNonRes}
40 898 --      {-----+      Some operation has taken place which precludes
40 899 --      | DoneTyping      further typing
40 900 --      +-----}
40 901 -- A  PROCEDURE TTEView.DoneTyping;
40 902 0- A  BEGIN
40 903 --      IF fTypingCommand <> NIL THEN
40 904 --          fTypingCommand.CompleteTyping;
40 905 -0 A  END;
40 906 --
40 907 --      {$S TERes}
40 908 --      {-----+
40 909 --      | DoSetCursor      |
40 910 --      +-----}
40 911 -- A  FUNCTION TTEView.DoSetCursor(localPoint: Point;
40 912 --      cursorRgn: RgnHandle): BOOLEAN; OVERRIDE;
40 913 --      VAR
40 914 --          qdExtent: Rect;
40 915 --
40 916 0- A  BEGIN
40 917 --      IF SELF <> gClipView THEN
40 918 1-      BEGIN
40 919 --          UserROMMap(TRUE);
40 920 --          SetCursor(GetCursor(iBeamCursor)^);
40 921 --          GetQDExtent(qdExtent);
40 922 --          RectRgn(cursorRgn, qdExtent);
40 923 --          DoSetCursor := TRUE;
40 924 -1      END
40 925 --      ELSE
40 926 --          DoSetCursor := FALSE;
40 927 -0 A  END;
40 928 --
40 929 --
40 930 --      {$S TEPrint}
40 931 --      {-----+      Get rid of the on-screen margin (fMargin) when
40 932 --      | DoSetPageOffset      printing by adjusting view
40 933 --      +-----}
40 934 -- A  PROCEDURE TTEView.DoSetPageOffset(coord: VPoint); OVERRIDE;
40 935 --      VAR
40 936 --          vhs: VHSelect;
40 937 --
40 938 0- A  BEGIN
40 939 --      INHERITED DoSetPageOffset(coord);
40 940 --      FOR vhs := v to h DO
40 941 --          IF coord.vh[vhs] = 0 THEN
40 942 --              gPageOffset.vh[vhs] := gPageOffset.vh[vhs] + fInset.topLeft.vh[vhs];
40 943 -0 A  END;
40 944 --
40 945 --
40 946 --      {$S TERes}
40 947 --      {-----+
40 948 --      | DoSetupMenus      |
40 949 --      +-----}
40 950 -- A  PROCEDURE TTEView.DoSetupMenus; OVERRIDE;
40 951 --      VAR
40 952 --          manyChars: BOOLEAN;
40 953 --
40 954 0- A  BEGIN
40 955 --      INHERITED DoSetupMenus;
40 956 --
40 957 --      WITH fHTE^ DO
40 958 --          manyChars := selStart < selEnd;
40 959 --      IF NOT MemSpaceIsLow THEN
40 960 1-      BEGIN
40 961 --          IF fAcceptsChanges THEN      ( One way or another, we can paste text )
40 962 --              CanPaste('TEXT');      ( If styles exist, all the better )
40 963 --
40 964 --          Enable(cCopy, manyChars);
40 965 -1      END;
40 966 --      Enable(cSelectAll, (fHTE^.teLength > 0));
40 967 --
40 968 --      { We enable Cut even if space is low, since it's nice to be able to rescue some of
40 969 --        the stuff you have to delete even if space is low. Note that it is possible to
40 970 --        get into the "can't do any commands" situation as a result. You should be able
40 971 --        to close and save the big document, then save the rescued text elsewhere, however. }
40 972 --
40 973 --      Enable(cCut, manyChars AND fAcceptsChanges);
40 974 --      Enable(cClear, manyChars AND fAcceptsChanges);
40 975 -0 A  END;
40 976 --
40 977 --
40 978 --      {$S TERes}
40 979 --      {-----+
40 980 --      | Draw      |
40 981 --      +-----}
40 982 -- A  PROCEDURE TTEView.Draw(area: Rect); OVERRIDE;
40 983 --      VAR
40 984 --          savedPort: GrafPtr;
40 985 --          aPort: GrafPtr;
40 986 --          isActive: BOOLEAN;
40 987 0- A  BEGIN
40 988 --      IF gPrinting OR gDrawingPictScrap THEN
40 989 1-      BEGIN

```





```

40 1089 -- B      PROCEDURE Hd1GivePasteFailed(error: OSErr; message: LONGINT);
40 1090 0- B      BEGIN
40 1091 --          IF aHandle <> NIL THEN
40 1092 1-              BEGIN
40 1093 --                  HUnlock(aHandle);
40 1094 --                  DisposHandle(aHandle);
40 1095 -1              END;
40 1096 --          IF error = phStylesTooBig THEN
40 1097 1-              BEGIN
40 1098 --                  StdAlert(phStylesTooBig);
40 1099 --                  GivePasteData := noErr;
40 1100 -1              END
40 1101 --          ELSE
40 1102 --              GivePasteData := error;
40 1103 -0 B      END;
40 1104 --
40 1105 0- A      BEGIN
40 1106 --          savedPerm := FALSE;
40 1107 --          aSize := 0;                                { Assume the worst }
40 1108 --          aHandle := NIL;
40 1109 --          CatchFailures(fi, Hd1GivePasteFailed);
40 1110 --
40 1111 --          IF dataType = 'TEXT' THEN
40 1112 1-              BEGIN
40 1113 --                  aSize := GetHandleSize(fText);
40 1114 --                  IF aDataHandle <> NIL THEN
40 1115 2-                      BEGIN
40 1116 --                          savedPerm := PermAllocation(TRUE);    { Passed in handle remains a permanent one }
40 1117 --                          SetHandleSize(aDataHandle, aSize);
40 1118 --                          err := MemError;                        { Save MemError & test it later because }
40 1119 --                          savedPerm := PermAllocation(savedPerm); { _PermAllocation may change it. }
40 1120 --                          IF err <> noErr THEN
40 1121 --                              Failure(err, 0);
40 1122 --                              BlockMove(fText^, aDataHandle^, aSize);
40 1123 -2                          END;
40 1124 -1                      END
40 1125 --                  ELSE IF dataType = 'styl' THEN
40 1126 1-                      BEGIN
40 1127 --                          IF fStyleType = kWithStyle THEN
40 1128 --                              IF NOT SpaceForStyles(0, MAXINT) THEN
40 1129 --                                  Failure(noErr, 0)                { We'll accept this error in worst case }
40 1130 --                              ELSE
40 1131 2-                                  BEGIN
40 1132 --                                      WITH fhTE^^ DO
40 1133 3-                                          BEGIN
40 1134 --                                              oldStart := selStart;
40 1135 --                                              oldEnd := selEnd;
40 1136 -3                                          END;
40 1137 --                                              SetSelect(0, MAXINT, fhTE);
40 1138 --                                              aHandle := Handle(GetStylScrap(fhTE));
40 1139 --                                              SetSelect(oldStart, oldEnd, fhTE);
40 1140 --
40 1141 --                                              IF aHandle <> NIL THEN
40 1142 3-                                                  BEGIN
40 1143 --                                                      aSize := GetHandleSize(aHandle);
40 1144 --                                                      IF aDataHandle <> NIL THEN
40 1145 4-                                                          BEGIN
40 1146 --                                                              savedPerm := PermAllocation(TRUE);
40 1147 --                                                              MoveHHi(aHandle);    { Try to prevent fragmentation, in case }
40 1148 --                                                              HLock(aHandle);    { Can't move while we're copying it! }
40 1149 --
40 1150 --                                                              err := PtrToXHand(aHandle^, { Copy styles into user-supplied handle }
40 1151 --                                                              aDataHandle, aSize);
40 1152 --                                                              HUnlock(aHandle);    { Okay for it to move again }
40 1153 --                                                              savedPerm := PermAllocation(savedPerm);
40 1154 --                                                              IF err <> noErr THEN    { Maybe enough for one copy, but not two! }
40 1155 --                                                                  Failure(phStylesTooBig, 0);
40 1156 -4                                                              END;
40 1157 --                                                              DisposHandle(aHandle);
40 1158 --                                                              aHandle := NIL;
40 1159 -3                                                          END
40 1160 --                                                      ELSE IF aDataHandle <> NIL THEN    { Hmm. There _was_ enough memory, but the }
40 1161 --                                                          Failure(phStylesTooBig, 0);    { _heap is probably pretty fragmented }
40 1162 --                                                      END;
40 1163 -2                                                              END
40 1164 -1                                  END
40 1165 --                              ELSE
40 1166 --                                  Failure(noTypeErr, 0);
40 1167 --
40 1168 --                                  FailSpaceIsLow;
40 1169 --                                  Success(fi);
40 1170 --                                  GivePasteData := aSize;
40 1171 -0 A      END;
40 1172 --
40 1173 --
40 1174 --          {$S TENonRes}
40 1175 --          {-----+
40 1176 --          |   InstallSelection
40 1177 --          +-----}
40 1178 -- A      PROCEDURE TTEView.InstallSelection(wasActive, beActive: BOOLEAN); OVERRIDE;
40 1179 --
40 1180 0- A      BEGIN
40 1181 --          IF Focus THEN ;                                { Try to focus because TEActivate may }
40 1182 --                                                        { _erase the caret. }
40 1183 --          IF beActive THEN
40 1184 1-              BEGIN
40 1185 --                  IF gConfiguration.hasScriptManager THEN
40 1186 --                      SetKeyScript(Font2Script(fTextStyle.tsFont));
40 1187 --                      TEActivate(fhTE);

```

```

40 1188 --      gCurrTEView := SELF;
40 1189 -1      END
40 1190 --      ELSE
40 1191 1-      BEGIN
40 1192 --      TEDeactivate(fHTE);
40 1193 --      DoneTyping;
40 1194 --      fSpecsChanged := TRUE;
40 1195 -1      END;
40 1196 -0 A  END;
40 1197 --
40 1198 --      {$S TEOpen}
40 1199 --      {-----+
40 1200 --      |  MakeTERecord          |  Call this after grafPort is properly set
40 1201 --      +-----}
40 1202 -- A  PROCEDURE TTEView.MakeTERecord;
40 1203 --      VAR
40 1204 --          anHTE:    TEHandle;
40 1205 --          dest:      Rect;
40 1206 --          oldPort:   GrafPtr;
40 1207 --          fi:        FailInfo;
40 1208 --
40 1209 -- B      PROCEDURE HdlMakeFailed(error: OSErr; message: LONGINT);
40 1210 0- B      BEGIN
40 1211 --          Free;
40 1212 -0 B      END;
40 1213 --
40 1214 --
40 1215 0- A  BEGIN
40 1216 --      GetPort(oldPort);
40 1217 --      SetPort(gWorkPort);
40 1218 --      SetPortTextStyle(fTextStyle);
40 1219 --
40 1220 --      dest.topLeft := fInset.topLeft;
40 1221 --      dest.right := fSize.h - fInset.right;
40 1222 --      dest.bottom := fSize.v - fInset.bottom;
40 1223 --
40 1224 --      IF fStyleType = kWithStyle THEN
40 1225 --          anHTE := TESTylNew(dest, dest)      ( Open a styled record if requested )
40 1226 --      ELSE
40 1227 --          anHTE := TENew(dest, dest);         ( _otherwise, do it the old way )
40 1228 --
40 1229 --      SetPort(oldPort);
40 1230 --
40 1231 --      CatchFailures(fi, HdlMakeFailed);      ( In case we couldn't create the TEHandle. )
40 1232 --      FailNIL(anHTE);                        ( Make sure we actually created one )
40 1233 --      fHTE := anHTE;                         ( We did, so save off TE handle )
40 1234 --      SetJustification(fJustification,
40 1235 --          kDontRedraw);                      ( Set justification to requested value )
40 1236 --      ChangeWrap(fAutoWrap, FALSE);          ( Install auto wrap (or CR only) )
40 1237 --      FailNoReserve;                          ( Got to have some reserve tank )
40 1238 --      Success(fi);
40 1239 --
40 1240 --      BeInPort(GetGrafPort);                  ( Associate with real port )
40 1241 -0 A  END;
40 1242 --
40 1243 --      {$S TERes}
40 1244 --      {-----+
40 1245 --      |  RecalcText          |  Pretty simple, anymore
40 1246 --      +-----}
40 1247 -- A  PROCEDURE TTEView.RecalcText;
40 1248 --
40 1249 0- A  BEGIN
40 1250 --      TECalcText(fHTE);
40 1251 -0 A  END;
40 1252 --
40 1253 --
40 1254 --      {$S TENonRes}
40 1255 --      {-----+
40 1256 --      |  Resize          |
40 1257 --      +-----}
40 1258 -- A  PROCEDURE TTEView.Resize (width, height: VCoordinate; invalidate: BOOLEAN); OVERRIDE;
40 1259 --      VAR
40 1260 --          needCalText:  BOOLEAN;
40 1261 --          r:            Rect;
40 1262 --          oldSize:      VPoint;
40 1263 0- A  BEGIN
40 1264 --          oldSize := fSize;
40 1265 --          INHERITED Resize(width, height, invalidate);
40 1266 --          IF fHTE <> NIL THEN
40 1267 1-      BEGIN
40 1268 --              r.topLeft := fInset.topLeft;
40 1269 --              r.right := fSize.h - fInset.right;
40 1270 --              r.bottom := fSize.v - fInset.bottom;
40 1271 --
40 1272 --              needCalText := (r.right <> fHTE^.destRect.right);
40 1273 --              StuffTERects(r);
40 1274 --              IF needCalText THEN
40 1275 2-      BEGIN
40 1276 --                  RecalcText;
40 1277 --                  SynchView(kDontRedraw);
40 1278 --                  IF invalidate |
40 1279 --                      (fAutoWrap AND (fSize.h <> oldSize.h) AND (fSize.v <> oldSize.v)) THEN
40 1280 --                      ForceRedraw;
40 1281 -2      END;
40 1282 -1      END;
40 1283 -0 A  END;
40 1284 --
40 1285 --
40 1286 --      {$S TERes}

```

```

40 1287 --      {-----+
40 1288 --      | ScrollSelectionIntoView |
40 1289 --      +-----+
40 1290 -- A  PROCEDURE TTEView.ScrollSelectionIntoView;
40 1291 --      CONST
40 1292 --          kMinAhead = 96;
40 1293 --      VAR
40 1294 --          selLoc:      Point;
40 1295 --          vhs:         VHSelect;
40 1296 --          newLoc:      Point;
40 1297 --          newSelLoc:   Point;
40 1298 --          newSelRect:  Rect;
40 1299 --          minToSee:    Point;
40 1300 --          visRect:     Rect;
40 1301 --          vNewSelRect: VRect;
40 1302 --
40 1303 0- A  BEGIN
40 1304 --      IF (fScroller <> NIL) & Focus THEN      ( Can't scroll selection if we don't have )
40 1305 1-      BEGIN                                  { -- a scroller! }
40 1306 --          GetVisibleRect(visRect);
40 1307 --          CalcSelLoc(newSelRect);
40 1308 --      ($IFC qDebug)
40 1309 --          IF gIntenseDebugging THEN
40 1310 2-      BEGIN
40 1311 --              WRITE('Incoming SelLoc: '); WritePt(selLoc); WRITELN;
40 1312 --              WRITE('Visible Rect was: '); WriteRect(visRect);
40 1313 --              WRITE('; Sel Rect was: '); WriteRect(newSelRect); WRITELN;
40 1314 -2      END;
40 1315 --      ($ENDC)
40 1316 --
40 1317 --      IF NOT RectsNest(visRect, newSelRect) THEN
40 1318 2-      BEGIN
40 1319 --          SetPt(minToSee,      ( Scroll the selection into view. )
40 1320 --              MIN(kMinAhead, fSize.h-newSelRect.left),
40 1321 --              LengthRect(newSelRect, v));
40 1322 --      ($IFC qDebug)
40 1323 --          IF gIntenseDebugging THEN
40 1324 3-      BEGIN
40 1325 --              WritelnRect('RevealRect: r', newSelRect);
40 1326 --              WritelnPt(' minToSee', minToSee);
40 1327 --              WRITELN;
40 1328 -3      END;
40 1329 --      ($ENDC)
40 1330 --          QDToViewRect(newSelRect, vNewSelRect);
40 1331 --          RevealRect(vNewSelRect, minToSee, kRedraw);
40 1332 --          IF Focus THEN ; ( Refocus in newly-scrolled position )
40 1333 -2      END;
40 1334 -1      END
40 1335 --      ELSE
40 1336 --      ($IFC NOT qNeedsROM128K)
40 1337 --          IF gConfiguration.hasROM128K THEN
40 1338 --      ($ENDC)
40 1339 --          IF NOT fAutoWrap & (fHTE <> NIL) THEN
40 1340 --              TESelView(fHTE);
40 1341 0- A  END;
40 1342 --
40 1343 --
40 1344 --      ($$ TENonRes)
40 1345 --      {-----+
40 1346 --      | SetJustification |
40 1347 --      +-----+
40 1348 -- A  PROCEDURE TTEView.SetJustification(newJust: INTEGER; redraw: BOOLEAN);
40 1349 0- A  BEGIN
40 1350 --      TSESetJust(newJust, fHTE);
40 1351 --      fJustification := newJust;
40 1352 --      IF redraw THEN
40 1353 --          ForceRedraw;
40 1354 0- A  END;
40 1355 --
40 1356 --      ($$ TENonRes)
40 1357 --      {-----+
40 1358 --      | SetOneStyle |
40 1359 --      +-----+
40 1360 -- A  PROCEDURE TTEView.SetOneStyle (theStart, theEnd, theMode: INTEGER; theStyle: TextStyle; redraw: BOOLEAN);
40 1361 --      VAR
40 1362 --          saveStart:  INTEGER;
40 1363 --          saveEnd:    INTEGER;
40 1364 --          savedPort:  GrafPtr;
40 1365 --          fInfo:      FontInfo;
40 1366 --          newStyle:   TextStyle;
40 1367 --
40 1368 0- A  BEGIN
40 1369 --      gFocusedView := NIL; ( ??? THIS SHOULDN'T BE NECESSARY! ??? )
40 1370 --      IF Focus THEN ;
40 1371 --      IF fStyleType = kWithStyle THEN
40 1372 1-      BEGIN
40 1373 --          WITH fHTE^^ DO
40 1374 2-      BEGIN
40 1375 --          saveStart := selStart;
40 1376 --          saveEnd := selEnd;
40 1377 -2      END;
40 1378 --          SetSelect(theStart, theEnd, fHTE);
40 1379 --          TSESetStyle(theMode, theStyle, redraw, fHTE);
40 1380 --          SetSelect(saveStart, saveEnd, fHTE);
40 1381 -1      END
40 1382 --      ELSE
40 1383 1-      BEGIN
40 1384 --          IF theMode = doAll THEN
40 1385 --              newStyle := theStyle

```

```

40 1386 --      ELSE
40 1387 --      BEGIN
40 1388 --          newStyle := fTextStyle;
40 1389 --          IF BAND(theMode, doFont) <> 0 THEN
40 1390 --              BEGIN
40 1391 --                  newStyle.tsFont := theStyle.tsFont;
40 1392 --                  IF gConfiguration.hasScriptManager THEN { _if Script Mgr is installed, change }
40 1393 --                      KeyScript(Font2Script(newStyle.tsFont)); { _keybd input system to match new font }
40 1394 --              END;
40 1395 --          IF BAND(theMode, doFace) <> 0 THEN
40 1396 --              newStyle.tsFace := theStyle.tsFace;
40 1397 --          IF BAND(theMode, doColor) <> 0 THEN
40 1398 --              newStyle.tsColor := theStyle.tsColor;
40 1399 --          IF BAND(theMode, addSize) <> 0 THEN
40 1400 --              newStyle.tsSize := newStyle.tsSize + theStyle.tsSize
40 1401 --          ELSE IF BAND(theMode, doSize) <> 0 THEN
40 1402 --              newStyle.tsSize := theStyle.tsSize;
40 1403 --          END;
40 1404 --
40 1405 --          GetPort(savedPort);
40 1406 --          SetPort(gWorkPort);
40 1407 --          SetPortTextStyle(newStyle);
40 1408 --          GetFontInfo(fInfo); { Need to get font's height and ascent. }
40 1409 --          SetPort(savedPort);
40 1410 --
40 1411 --          WITH fhTE^^, newStyle, fInfo DO
40 1412 --              BEGIN
40 1413 --                  txSize := tsSize;
40 1414 --                  txFont := tsFont;
40 1415 --                  txFace := tsFace;
40 1416 --                  fontAscent := ascent;
40 1417 --                  lineHeight := ascent + descent + leading;
40 1418 --                  SetIfColor(tsColor);
40 1419 --              END;
40 1420 --          fTextStyle := newStyle;
40 1421 --      END;
40 1422 --
40 1423 --      IF TRUE ( (fStyleType = kWithoutStyle) OR (theStart <> theEnd) ) THEN
40 1424 --          BEGIN
40 1425 --              RecalcText;
40 1426 --              SynchView(redraw AND (fStyleType = kWithStyle));
40 1427 --              IF redraw AND (fStyleType = kWithoutStyle) THEN
40 1428 --                  ForceRedraw;
40 1429 --              END;
40 1430 --              fSpecsChanged := TRUE;
40 1431 --          END;
40 1432 --
40 1433 --      {$S TENonRes}
40 1434 --      {-----+
40 1435 --      | SetText | Convenience routine, to avoid creating a handle
40 1436 --      +-----+
40 1437 --      A PROCEDURE TTEView.SetText(theText: Str255);
40 1438 --      VAR
40 1439 --          theTextHandle: Handle;
40 1440 --      BEGIN
40 1441 --          IF fhTE <> NIL THEN { If we're replacing text, styles are kaput }
40 1442 --              BEGIN
40 1443 --                  FailOSErr(PtrToHand(@theText[1], theTextHandle, LENGTH(theText)));
40 1444 --                  StuffText(theTextHandle);
40 1445 --              END;
40 1446 --          END;
40 1447 --
40 1448 --      {$S TENonRes}
40 1449 --      {-----+
40 1450 --      | ShowReverted |
40 1451 --      +-----+
40 1452 --      A PROCEDURE TTEView.ShowReverted; OVERRIDE;
40 1453 --      BEGIN
40 1454 --          RecalcText;
40 1455 --          fLastHeight := 0;
40 1456 --          INHERITED ShowReverted;
40 1457 --      END;
40 1458 --
40 1459 --
40 1460 --      {$S TENonRes}
40 1461 --      {-----+ Return TRUE if there is enough memory to hold the
40 1462 --      | SpaceForStyles | styles associated with the given range of text
40 1463 --      +-----+
40 1464 --      A FUNCTION TTEView.SpaceForStyles(rangeStart, rangeEnd: LONGINT): BOOLEAN;
40 1465 --      VAR
40 1466 --          h: Handle;
40 1467 --      BEGIN
40 1468 --          h := NewPermHandle(TENumStyles(rangeStart, rangeEnd, fhTE) * SIZEOF(ScrpSTElement) + 2);
40 1469 --          IF (h = NIL) THEN
40 1470 --              BEGIN
40 1471 --                  StdAlert(phStylesTooBig);
40 1472 --                  SpaceForStyles := FALSE;
40 1473 --              END
40 1474 --          ELSE
40 1475 --              BEGIN
40 1476 --                  DisposHandle(h); { Release memory back to the system }
40 1477 --                  SpaceForStyles := TRUE;
40 1478 --              END;
40 1479 --      END;
40 1480 --
40 1481 --      {$S TENonRes}
40 1482 --      {-----+
40 1483 --      | StuffStyles |
40 1484 --      +-----+

```



```

40 1584 --      fLastHeight := theHeight;                ( Remember new height value )
40 1585 --      END;
40 1586 --
40 1587 --      IF redraw THEN
40 1588 --          { First, make sure selection is visible (this conveniently focuses). Then,
40 1589 --            repair any extra feedback which TextEdit may have mashed. }
40 1590 --          ScrollSelectionIntoView;
40 1591 -- A      END;
40 1592 --
40 1593 --
40 1594 --      {$S TENonRes}
40 1595 --      {-----+
40 1596 --      | WriteToDeskScrap
40 1597 --      +-----}
40 1598 -- A      PROCEDURE TTEView.WriteToDeskScrap; OVERRIDE;
40 1599 --      VAR
40 1600 --          aHandle:   Handle;
40 1601 --
40 1602 -- A      BEGIN
40 1603 --          FailOSErr(PutDeskScrapData('TEXT', fText));
40 1604 --
40 1605 --          IF (fStyleType = kWithStyle) & SpaceForStyles(0, MAXINT) THEN
40 1606 --              BEGIN
40 1607 --                  SetSelect(0, MAXINT, fHTE);
40 1608 --                  aHandle := Handle(GetStylScrap(fHTE));
40 1609 --                  FailNIL(aHandle);
40 1610 --                  FailOSErr(PutDeskScrapData('styl', aHandle));
40 1611 --              END;
40 1612 -- A      END;
40 1613 --
40 1614 --
40 1615 --      {$IFC qDebug}
40 1616 --      {$S TDEbug}
40 1617 --      {-----+
40 1618 --      | IdentifySoftware
40 1619 --      +-----}
40 1620 -- A      PROCEDURE TTEView.IdentifySoftware; OVERRIDE;
40 1621 -- A      BEGIN
40 1622 --          WRITELN('UTEView of 13Nov87, Compiled on ', COMPDATE, ' @ ', COMPTIME);
40 1623 --          INHERITED IdentifySoftware;
40 1624 -- A      END;
40 1625 --      {$ENDC}
40 1626 --
40 1627 --
40 1628 --      {$IFC qDebug}
40 1629 --      {$S TEFIELDS}
40 1630 --      {-----+
40 1631 --      | Fields
40 1632 --      +-----}
40 1633 -- A      PROCEDURE TTEView.Fields (PROCEDURE DoToField (fieldName: Str255;
40 1634 --                                     fieldAddr: Ptr;
40 1635 --                                     fieldType: INTEGER)); OVERRIDE;
40 1636 --
40 1637 -- A      BEGIN
40 1638 --          DoToField('TTEView', NIL, bClass);
40 1639 --          DoToField('fHTE', @fHTE, bTEHandle);
40 1640 --          DoToField('fText', @fText, bHandle);
40 1641 --          DoToField('fSavedTEHandle', @fSavedTEHandle, bTEHandle);
40 1642 --          DoToField('fInset', @fInset, bRect);
40 1643 --          DoToField('fKeyCmdNumber', @fKeyCmdNumber, bCmdNumber);
40 1644 --          DoToField('fMaxChars', @fMaxChars, bInteger);
40 1645 --          DoToField('fLastHeight', @fLastHeight, bInteger);
40 1646 --          DoToField('fTypingCommand', @fTypingCommand, bObject);
40 1647 --          {$H-}
40 1648 --          TextStyleFields('fTextStyle', fTextStyle, DoToField);
40 1649 --          {$H+}
40 1650 --          DoToField('fJustification', @fJustification, bInteger);
40 1651 --          DoToField('fAcceptsChanges', @fAcceptsChanges, bBoolean);
40 1652 --          DoToField('fStyleType', @fStyleType, bBoolean);
40 1653 --          DoToField('fAutoWrap', @fAutoWrap, bBoolean);
40 1654 --          DoToField('fFreeText', @fFreeText, bBoolean);
40 1655 --          DoToField('fSpecsChanged', @fSpecsChanged, bBoolean);
40 1656 --          DoToField('fScroller', @fScroller, bObject);
40 1657 --          DoToField('fLastLine', @fLastLine, bInteger);
40 1658 --          DoToField('fLastPageBreak', @fLastPageBreak, bInteger);
40 1659 --          INHERITED Fields(DoToField);
40 1660 -- A      END;
40 1661 --      {$ENDC}
38 625 --      {$I UTEView.TTECommand.p}

```

```

41 1 --      {$P}
41 2 --      {TEView.TTECommand.p}
41 3 --      {Copyright © 1984-1988 Apple Computer Inc. All rights reserved.}
41 4 --
41 5 --      (*****
41 6 --      (*      T T E C o m m a n d      *)
41 7 --      (*****
41 8 --
41 9 --      {$S TEselCommand}
41 10 --      {-----+
41 11 --      |      ITECommand      |
41 12 --      +-----}
41 13 -- A  PROCEDURE TTECommand(itsTEView: TTEView; itsCmdNumber: CmdNumber; itsSaveText: BOOLEAN);
41 14 --      VAR
41 15 --          selChars: INTEGER;
41 16 --          h: Handle;
41 17 --          fi: FailInfo;
41 18 --
41 19 --          {-----+
41 20 --          |      HdInitFailed      |
41 21 --          +-----}
41 22 -- B  PROCEDURE HdInitFailed(error: OSErr; message: LONGINT);
41 23 -- B  BEGIN
41 24 --      Free;
41 25 -- B  END;
41 26 --
41 27 -- A  BEGIN
41 28 --      fTEView := itsTEView;
41 29 --      fhTE := itsTEView.fhTE;
41 30 --
41 31 --      WITH fhTE^^ DO
41 32 --      BEGIN
41 33 --          foldStart := selStart;
41 34 --          foldEnd := selEnd;
41 35 --          selChars := selEnd - selStart;
41 36 --      END;
41 37 --
41 38 --      foldText := NIL;
41 39 --      foldStyles := NIL;
41 40 --
41 41 --      fNewStart := 0;
41 42 --      fNewEnd := 0;
41 43 --      fNewText := NIL;
41 44 --      fNewStyles := NIL;
41 45 --
41 46 --      fPadding := NIL;
41 47 --      fTextPad := 0;
41 48 --      fStylePad := 0;
41 49 --
41 50 --      ICommand(itsCmdNumber, itsTEView.fDocument, NIL, NIL);
41 51 --      CatchFailures(fi, HdInitFailed);
41 52 --
41 53 --      IF itsSaveText THEN
41 54 --      BEGIN
41 55 --          h := NewPermHandle(selChars);
41 56 --          FailNIL(h);
41 57 --
41 58 --          IF selChars > 0 THEN
41 59 --              BlockMove(Pointer(ORD(fhTE^^.hText^) + foldStart), h^, selChars);
41 60 --
41 61 --              foldText := h;
41 62 --              fTextPad := foldStart - foldEnd;
41 63 --              fPadding := NewPermHandle(0);
41 64 --              FailNIL(fPadding);
41 65 --      END;
41 66 --
41 67 --      { TextEdit has this "feature" which it exercises if it runs out of memory. It's
41 68 --      called DS number 25. We'll try to avoid it by assuring that enough memory exists
41 69 --      to fulfill the request, but we won't die because of it. This is a particularly
41 70 --      ugly situation - there could be >600K of style information associated with a 32K
41 71 --      block of text. And to support undo, we've got to assume that there may momentarily
41 72 --      be THREE copies floating around, adding up to a total potential liability of almost
41 73 --      2 Meg for a single TE record. The worst that can happen, though, is that the text
41 74 --      will be safe, but it won't have any styles associated with it. }
41 75 --
41 76 --      IF (itsTEView.fstyleType = kWithStyle) &
41 77 --          itsTEView.SpaceForStyles(fhTE^^.selStart, fhTE^^.selEnd) THEN
41 78 --      BEGIN
41 79 --          foldStyles := GetStyleScrap(fhTE);
41 80 --          FailNIL(foldStyles);
41 81 --          fStylePad := GetHandleSize(Handle(foldStyles));
41 82 --      END;
41 83 --
41 84 --      Success(fi);
41 85 --
41 86 -- A  END;
41 87 --
41 88 --
41 89 --      {$S TEdoCommand}
41 90 --      {-----+
41 91 --      |      Free      |
41 92 --      +-----}
41 93 -- A  PROCEDURE TTECommand.Free; OVERRIDE;
41 94 -- A  BEGIN
41 95 --      DisposeIfHandle(foldText);
41 96 --      DisposeIfHandle(Handle(foldStyles));
41 97 --      DisposeIfHandle(fNewText);
41 98 --      DisposeIfHandle(Handle(fNewStyles));

```



```

41 99 --      DisposIfHandle(fPadding);
41 100 --
41 101 --      INHERITED Free;
41 102 -0 A  END;
41 103 --
41 104 --
41 105 --      {$S TEdoCommand}
41 106 --      {-----+
41 107 --      | BanishOldText
41 108 --      +-----}
41 109 -- A  PROCEDURE TTECommand.BanishOldText;
41 110 0- A  BEGIN
41 111 --      IF foldEnd > foldStart THEN
41 112 --          TDelete(fHTE);
41 113 --          SetHandleSize(fPadding, MAX(-(fTextPad+fStylePad), 0));
41 114 -0 A  END;
41 115 --
41 116 --
41 117 --      {$S TEdoCommand}
41 118 --      {-----+
41 119 --      | InstallNewText
41 120 --      +-----}
41 121 -- A  PROCEDURE TTECommand.InstallNewText;
41 122 --      VAR
41 123 --          savedSize: LONGINT;
41 124 --          itsText:   Handle;
41 125 --
41 126 0- A  BEGIN
41 127 --      IF fNewEnd > fNewStart THEN
41 128 1-          BEGIN
41 129 --          itsText := fTEView.fText;
41 130 --          savedSize := GetHandleSize(itsText);
41 131 --
41 132 --          {$IFC qDebug}
41 133 --          IF fNewText = NIL THEN
41 134 --              ProgramBreak('InstallNewText called with fNewText = NIL!');
41 135 --          {$ENDC}
41 136 --
41 137 --          MoveHHI(fNewText);
41 138 --          HLock(fNewText);
41 139 --
41 140 --          IF fTEView.fStyleType = kWithStyle THEN { If record has style, use it }
41 141 --              TEstylInsert(fNewText^, ( (It's okay for fNewStyles to be NIL here) }
41 142 --              GetHandleSize(fNewText), fNewStyles, fHTE)
41 143 --          ELSE { Otherwise, do it the old-fashioned way }
41 144 --              TEInsert(fNewText^, GetHandleSize(fNewText), fHTE);
41 145 --
41 146 --          HUnlock(fNewText);
41 147 --
41 148 --          IF GetHandleSize(itsText) <= savedSize THEN
41 149 --              FailOSErr(memFullErr);
41 150 --
41 151 --          fTEView.fSpecsChanged := TRUE;
41 152 -1  END;
41 153 -0 A  END;
41 154 --
41 155 --
41 156 --      {$IFC qDebug}
41 157 --      {$S TFields}
41 158 --      {-----+
41 159 --      | Fields
41 160 --      +-----}
41 161 -- A  PROCEDURE TTECommand.Fields (PROCEDURE DoToField (fieldName: Str255;
41 162 --          fieldAddr: Ptr;
41 163 --          fieldType: INTEGER)); OVERRIDE;
41 164 0- A  BEGIN
41 165 --      DoToField('TTECommand', NIL, bClass);
41 166 --      DoToField('fTEView', @fTEView, bObject);
41 167 --      DoToField('fHTE', @fHTE, bTEHandle);
41 168 --      DoToField('foldStart', @fOldStart, bInteger);
41 169 --      DoToField('foldEnd', @fOldEnd, bInteger);
41 170 --      DoToField('foldText', @fOldText, bHandle);
41 171 --      DoToField('foldStyles', @fOldStyles, bHandle);
41 172 --      DoToField('fNewStart', @fNewStart, bInteger);
41 173 --      DoToField('fNewEnd', @fNewEnd, bInteger);
41 174 --      DoToField('fNewText', @fNewText, bHandle);
41 175 --      DoToField('fNewStyles', @fNewStyles, bHandle);
41 176 --      DoToField('fPadding', @fPadding, bHandle);
41 177 --      DoToField('fTextPad', @fTextPad, bInteger);
41 178 --      DoToField('fStylePad', @fStylePad, bLongInt);
41 179 --      INHERITED Fields(DoToField);
41 180 -0 A  END;
41 181 --      {$ENDC}
41 182 --
41 183 --
41 184 --      {$S TEdoCommand}
41 185 --      {-----+
41 186 --      | RemoveAdditions
41 187 --      +-----}
41 188 -- A  PROCEDURE TTECommand.RemoveAdditions;
41 189 0- A  BEGIN
41 190 --      IF fNewText <> NIL THEN
41 191 1-          BEGIN
41 192 --          TSetSelect(fNewStart, fNewEnd, fHTE);
41 193 --          TDelete(fHTE);
41 194 -1  END;
41 195 --          SetHandleSize(fPadding, MAX(fTextPad+fStylePad, 0));
41 196 -0 A  END;
41 197 --

```

```

41 198 --
41 199 --      {$S TEDoCommand}
41 200 --      {-----+
41 201 --      |   RestoreSelection   |
41 202 --      +-----+
41 203 -- A  PROCEDURE TTECommand.RestoreSelection;
41 204 0- A  BEGIN
41 205 --      TSEtSelect(foldStart, foldEnd, fhTE);
41 206 0 A  END;
41 207 --
41 208 --
41 209 --      {$S TEDoCommand}
41 210 --      {-----+
41 211 --      |   ReviveDeletions   |   Used for UNDO phase
41 212 --      +-----+
41 213 -- A  PROCEDURE TTECommand.ReviveDeletions;
41 214 --      VAR
41 215 --          itsText:      Handle;
41 216 --          savedSize:    LONGINT;
41 217 --          nChars:      INTEGER;
41 218 0- A  BEGIN
41 219 --      TSEtSelect(foldStart, foldStart, fhTE);   { so insert will take place at right point }
41 220 --      nChars := foldEnd - foldStart;
41 221 --      IF nChars > 0 THEN
41 222 1-      BEGIN
41 223 --          itsText := fTEView.fText;
41 224 --          savedSize := GetHandleSize(itsText);
41 225 --
41 226 --          MoveHHI(foldText);   { Prevent heap fragmentation }
41 227 --          HLock(foldText);
41 228 --
41 229 --          IF fTEView.fStyleType = kWithStyle THEN { If record has style, use it }
41 230 --              TESTylInsert(foldText~, nChars,   { (It's okay for foldStyles to be NIL here) }
41 231 --                  foldStyles, fhTE)
41 232 --          ELSE
41 233 --              TEInsert(foldText~, nChars, fhTE);   { Otherwise, do it the old-fashioned way }
41 234 --
41 235 --          HUnlock(foldText);
41 236 --
41 237 --          IF GetHandleSize(itsText) <= savedSize THEN
41 238 --              FailOSErr(memFullErr);
41 239 --
41 240 --          fTEView.fSpecsChanged := TRUE;
41 241 1-      END;
41 242 0 A  END;
41 243 --
41 244 --
41 245 --      {$S TEDoCommand}
41 246 --      {-----+
41 247 --      |   DoMainFunction   |   Used for DO and REDO
41 248 --      +-----+
41 249 -- A  PROCEDURE TTECommand.DoMainFunction;
41 250 0- A  BEGIN
41 251 --      IF fCmdNumber <> cCopy THEN
41 252 --          BanishOldText;
41 253 --          InstallNewText;
41 254 --      IF fCmdNumber <> cCopy THEN
41 255 --          fTEView.SynchView(kRedraw);
41 256 0 A  END;
41 257 --
41 258 --
41 259 --      {$S TEDoCommand}
41 260 --      {-----+
41 261 --      |   DoIt   |
41 262 --      +-----+
41 263 -- A  PROCEDURE TTECommand.DoIt; OVERRIDE;
41 264 0- A  BEGIN
41 265 --      IF fTEView.Focus THEN ;   {??? What if Focus fails}
41 266 --
41 267 --          DoMainFunction;
41 268 --      {$IFC qDebug}
41 269 --          IF gTEIntenseDebugging THEN
41 270 --              DumpTTECommand(SELF);
41 271 --      {$ENDC}
41 272 0 A  END;
41 273 --
41 274 --
41 275 --      {$S TEDoCommand}
41 276 --      {-----+
41 277 --      |   UndoIt   |
41 278 --      +-----+
41 279 -- A  PROCEDURE TTECommand.UndoIt; OVERRIDE;
41 280 0- A  BEGIN
41 281 --      IF fTEView.Focus THEN ;   {??? What if Focus fails}
41 282 --
41 283 --          RemoveAdditions;
41 284 --          ReviveDeletions;
41 285 --          RestoreSelection;
41 286 --      IF fCmdNumber <> cCopy THEN
41 287 --          fTEView.SynchView(kRedraw);
41 288 --      {$IFC qDebug}
41 289 --          IF gTEIntenseDebugging THEN
41 290 --              DumpTTECommand(SELF);
41 291 --      {$ENDC}
41 292 0 A  END;
41 293 --
41 294 --
41 295 --      {$S TEDoCommand}
41 296 --      {-----+

```

```

41 297 -- | RedoIt
41 298 -- +-----+
41 299 -- A PROCEDURE TTECommand.RedoIt: OVERRIDE;
41 300 0- A BEGIN
41 301 -- IF fTEView.Focus THEN ; (??? What if Focus fails)
41 302 --
41 303 -- RestoreSelection;
41 304 -- DoMainFunction;
41 305 -- (SIFC qDebug)
41 306 -- IF gTEIntenseDebugging THEN
41 307 -- DumpTTECommand(SELf);
41 308 -- ($ENDC)
41 309 0- A END;
41 310 --
41 311 --
41 312 -- (*****
41 313 -- (* T T E C u t C o p y C o m m a n d *)
41 314 -- (*****
41 315 --
41 316 -- ($S TESeICommand)
41 317 -- {-----+
41 318 -- | ITECutCopyCommand
41 319 -- +-----+
41 320 -- A PROCEDURE TTECutCopyCommand.ITECutCopyCommand(itsTEView: TTEView; itsCmdNumber: CmdNumber);
41 321 0- A BEGIN
41 322 -- fClipCreated := FALSE;
41 323 -- ITECommand(itsTEView, itsCmdNumber, TRUE);
41 324 -- fChangesClipboard := TRUE;
41 325 -- fCausesChange := itsCmdNumber <> cCopy;
41 326 0- A END;
41 327 --
41 328 --
41 329 -- ($S TEdoCommand)
41 330 -- {-----+
41 331 -- | Free
41 332 -- +-----+
41 333 -- A PROCEDURE TTECutCopyCommand.Free: OVERRIDE;
41 334 0- A BEGIN
41 335 -- IF fClipCreated THEN
41 336 -- fOldText := NIL;
41 337 -- INHERITED Free;
41 338 0- A END;
41 339 --
41 340 --
41 341 -- ($S TEdoCommand)
41 342 -- {-----+
41 343 -- | DoIt
41 344 -- +-----+
41 345 -- A PROCEDURE TTECutCopyCommand.DoIt: OVERRIDE;
41 346 -- VAR
41 347 -- clipTEView: TTEView;
41 348 -- clipHere: BOOLEAN;
41 349 -- fi: FailInfo;
41 350 -- clipStyle: TextStyle;
41 351 -- itsSize: VPoint;
41 352 -- itsMargins: Rect;
41 353 --
41 354 -- {-----+
41 355 -- | HdIClipFailed
41 356 -- +-----+
41 357 -- B PROCEDURE HdIClipFailed(error: OSerr; message: LONGINT);
41 358 0- B BEGIN
41 359 -- clipTEView.Free;
41 360 0- B END;
41 361 --
41 362 0- A BEGIN (TTECutCopyCommand.DoIt)
41 363 -- IF fTEView.Focus THEN ; (??? What if Focus fails)
41 364 --
41 365 -- SetTextStyle(clipStyle, applFont, [1, { Initial style same as virgin TEView }
41 366 -- 12, gRGBBlack);
41 367 --
41 368 -- SetVPt(itsSize, 100, 50); { An arbitrary initial size. }
41 369 -- SetRect(itsMargins, 10, 8, 10, 0); { No bottom margin. }
41 370 --
41 371 -- New(clipTEView); { Create a new view for the clipboard }
41 372 -- FailNIL(clipTEView);
41 373 -- WITH fTEView DO
41 374 -- clipTEView.ITEView(NIL, NIL, { Initialize view }
41 375 -- gZeroVPt, itsSize,
41 376 -- sizeSuperView, sizeVariable,
41 377 -- itsMargins, clipStyle,
41 378 -- teJustLeft, fStyleType, fAutoWrap);
41 379 -- clipTEView.fAcceptsChanges := FALSE; { This is a read-only view }
41 380 --
41 381 -- CatchFailures(fi, HdIClipFailed); { Cut can eat into temp memory so users can }
41 382 -- { _rescue text from overweight documents }
41 383 -- IF NOT fCausesChange THEN { If Copy-ing, assure there's enough room }
41 384 -- FailSpaceIsLow;
41 385 -- Success(fi);
41 386 -- clipTEView.StuffText(fOldText);
41 387 -- FailSpaceIsLow;
41 388 --
41 389 -- (??? GOT TO FIGURE OUT SOME WAY TO PRE-FLIGHT THIS! ????????????????????????????????????????? )
41 390 -- IF clipTEView.fStyleType = kWithStyle THEN { If record has style }
41 391 -- SetStylScrap(0, MAXINT, fOldStyles, { _then put in the styles }
41 392 -- FALSE, clipTEView.fHTE);
41 393 -- FailSpaceIsLow;
41 394 --
41 395 -- clipTEView.fFreeText := TRUE; { Let TEView know it has to free the text }

```

```

41 396 --
41 397 --      gApplication.ClaimClipboard(clipTEView);      ( Okay to claim (will call RecalcText!) )
41 398 --
41 399 --      fClipCreated := TRUE;                          ( We be done )
41 400 --      DoMainFunction;                                ( Do the actual cut/copy )
41 401 --
41 402 --      ($IFC qDebug)
41 403 --      IF gTEIntenseDebugging THEN
41 404 1-      BEGIN
41 405 --          DumpTTERecord(clipTEView.fHTE);
41 406 --          DumpTTECommand(SELF);
41 407 -1      END;
41 408 --      ($ENDC)
41 409 -0 A  END;
41 410 --
41 411 --
41 412 --      ($S TEDoCommand)
41 413 --      {-----+
41 414 --      |   ReviveDeletions   |
41 415 --      +-----}
41 416 --      PROCEDURE TTECutCopyCommand.ReviveDeletions; OVERRIDE;
41 417 0- A  BEGIN
41 418 --          IF fCmdNumber = cCut THEN
41 419 --              INHERITED ReviveDeletions;              ( Don't do it for COPY )
41 420 -0 A  END;
41 421 --
41 422 --
41 423 --      (*****
41 424 --      (*      T T E P a s t e C o m m a n d      *)
41 425 --      (*****
41 426 --
41 427 --      ($S TSElCommand)
41 428 --      {-----+
41 429 --      |   ITEPasteCommand   |
41 430 --      +-----}
41 431 -1 A  PROCEDURE TTEPasteCommand.ITEPasteCommand(itsTEView: TTEView);
41 432 --      ( We can't use TEPaste because it clobbers the DeskScrap: the text would be recoverable
41 433 --      from the special TextEdit Scrap, but other types of non-TEXT scrap are permanently
41 434 --      lost, it seems )
41 435 --      VAR
41 436 --          savedPerm:      BOOLEAN;
41 437 --          newLength:      INTEGER;
41 438 --          newStyleLen:    LONGINT;
41 439 --          newText:        Handle;
41 440 --          newStyles:      Handle;
41 441 --          dataType:       ResType;
41 442 --          fi:             FailInfo;
41 443 --
41 444 --      {-----+
41 445 --      |   Hd1PasteFailed   |
41 446 --      +-----}
41 447 -1 B  PROCEDURE Hd1PasteFailed(error: OSErr; message: LONGINT);
41 448 0- B  BEGIN
41 449 --          savedPerm := PermAllocation(savedPerm);
41 450 --          DisposIfHandle(newText);
41 451 --          DisposIfHandle(Handle(newStyles));
41 452 --          Free;
41 453 -0 B  END;
41 454 --
41 455 0- A  BEGIN
41 456 --          ITECommand(itsTEView, cPaste, TRUE);      ( Perform stock initializations )
41 457 --
41 458 --          savedPerm := FALSE;
41 459 --
41 460 --          newStyleLen := 0;                          ( Assume there are no new styles )
41 461 --          newStyles := NIL;
41 462 --          newText := NIL;
41 463 --
41 464 --          newText := NewPermHandle(0);                ( Create a handle to receive clipboard data )
41 465 --          FailNIL(newText);
41 466 --          IF itsTEView.fStyleType = kWithStyle THEN
41 467 1-      BEGIN
41 468 --              newStyles := NewPermHandle(0);          ( Same for handle to receive style info )
41 469 --              FailNIL(Handle(newStyles));
41 470 -1      END;
41 471 --
41 472 --          CatchFailures(fi, Hd1PasteFailed);
41 473 --
41 474 --          newLength := gApplication.GetDataToPaste(newText, dataType);
41 475 --
41 476 --          IF newLength > 0 THEN
41 477 1-      BEGIN
41 478 --          ($IFC qDebug)
41 479 --              IF dataType <> 'TEXT' THEN
41 480 --                  ProgramBreak('TEPasteCommand given some non-text from clipboard')
41 481 --              ELSE
41 482 --      ($ENDC)
41 483 2-      BEGIN
41 484 --                  ( Prime "new" values )
41 485 --                  fNewText := newText;
41 486 --                  fNewStart := fHTE^^.selStart;
41 487 --                  fNewEnd := fNewStart + newLength;
41 488 --                  fTextPad := newLength - (fOldEnd - fOldStart);
41 489 --
41 490 3-      IF itsTEView.fStyleType = kWithStyle THEN
41 491 --          BEGIN
41 492 --              newStyleLen := gClipView.GivePasteData(newStyles, 'styl');
41 493 4-      IF newStyleLen > 0 THEN
41 494 --          BEGIN
41 495 --              fNewStyles := StScrpHandle(newStyles);

```

```

41 495 --          fStylePad :=          { Difference between old and new styles }
41 496 --          newStyleLen - fStylePad;
41 497 --          END;
41 498 --          END;
41 499 --
41 500 --          savedPerm := PermAllocation(TRUE); { fPadding remains a permanent handle_ }
41 501 --          SetHandleSize(fPadding, MAX(fTextPad+fStylePad, 0));
41 502 --          FailMemError;
41 503 --
41 504 --          FailSpaceIsLow;
41 505 --          END;
41 506 --          END;
41 507 --          Success(fi);
41 508 --          savedPerm := PermAllocation(savedPerm); { Allocation state back to the way it was }
41 509 -- A      END;
41 510 --
41 511 --          (*****
41 512 --          (*      T T E S t y l e C o m m a n d
41 513 --          (*****
41 514 --
41 515 --          ($$ TEselCommand)
41 516 --          {-----+
41 517 --          |      I T E S t y l e C o m m a n d
41 518 --          +-----}
41 519 -- A      PROCEDURE TTEStyleCommand.ITEStyleCommand(itsTEView: TTEView; itsNewStyle: TextStyle;
41 520 --          itsCmdNumber: CmdNumber; itsMode: INTEGER);
41 521 --          VAR
41 522 --              savedPerm:    BOOLEAN;
41 523 --              fi:           FailInfo;
41 524 --
41 525 -- A      BEGIN
41 526 --
41 527 --          ITECommand(itsTEView, itsCmdNumber, FALSE); { Perform stock initialization, sans text }
41 528 --
41 529 --          fOldTextStyle := itsTEView.fTextStyle;
41 530 --          fNewTextStyle := itsNewStyle;
41 531 --
41 532 --          fMode := itsMode;
41 533 --          IF NOT gConfiguration.hasColorQD THEN
41 534 --              fMode := BAND(fMode, BNOT(doColor)); { Only do color change if we can }
41 535 -- A      END;
41 536 --
41 537 --          ($$ TEDoCommand)
41 538 --          {-----+
41 539 --          |      I n s t a l l O n e S t y l e
41 540 --          +-----}
41 541 -- A      PROCEDURE TTEStyleCommand.InstallOneStyle(newStyl: TextStyle);
41 542 --
41 543 -- A      BEGIN
41 544 --          fTEView.SetOneStyle(fOldStart, fOldEnd, fMode, newStyl, kRedraw); { Focus'es for us }
41 545 -- A      END;
41 546 --
41 547 --          ($$ TEDoCommand)
41 548 --          {-----+
41 549 --          |      I n s t a l l M a n y S t y l e s
41 550 --          +-----}
41 551 -- A      PROCEDURE TTEStyleCommand.InstallManyStyles(newStyls: StScrpHandle);
41 552 --
41 553 -- A      BEGIN
41 554 --          IF fTEView.Focus THEN :
41 555 --              { No need to check for fStyleType, since we only get here if the record is stylish }
41 556 --              SetStylScrap(fOldStart, fOldEnd, newStyls, TRUE, fHTE);
41 557 --              fTEView.RecalcText; { Might have changed number of lines }
41 558 --              fTEView.SynchView(kRedraw); { Show corrected view }
41 559 --
41 560 --              fTEView.fSpecsChanged := TRUE;
41 561 -- A      END;
41 562 --
41 563 --          ($$ TEDoCommand)
41 564 --          {-----+
41 565 --          |      D o I t
41 566 --          +-----}
41 567 -- A      PROCEDURE TTEStyleCommand.DoIt: OVERRIDE;
41 568 --
41 569 -- A      BEGIN
41 570 --          InstallOneStyle(fNewTextStyle);
41 571 --          fMode := BAND(fMode, BNOT(doToggle)); { Turn off toggle mode, if set }
41 572 --          ($IFC qDebug)
41 573 --              IF gTEIntenseDebugging THEN
41 574 --                  DumpTTECommand(SELF);
41 575 --          ($ENDC)
41 576 -- A      END;
41 577 --
41 578 --          ($$ TEDoCommand)
41 579 --          {-----+
41 580 --          |      U n d o I t
41 581 --          +-----}
41 582 -- A      PROCEDURE TTEStyleCommand.UndoIt: OVERRIDE;
41 583 --
41 584 -- A      BEGIN
41 585 --          IF fTEView.fStyleType = kWithoutStyle THEN
41 586 --              InstallOneStyle(fOldTextStyle)
41 587 --          ELSE
41 588 --              InstallManyStyles(fOldStyles);
41 589 --          ($IFC qDebug)
41 590 --              IF gTEIntenseDebugging THEN
41 591 --                  DumpTTECommand(SELF);
41 592 --          ($ENDC)
41 593 -- A      END;

```

```

41 594 --
41 595 --      ($$ TEdoCommand)
41 596 --      {-----+
41 597 --      |      Redoit
41 598 --      +-----+
41 599 -- A  PROCEDURE TTEStyleCommand.Redoit; OVERRIDE;
41 600 --
41 601 0- A  BEGIN
41 602 --      DoIt;
41 603 0- A  END;
41 604 --
41 605 --      ($IFC qDebug)
41 606 --      ($$ TEFields)
41 607 --      {-----+
41 608 --      |      Fields
41 609 --      +-----+
41 610 -- A  PROCEDURE TTEStyleCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
41 611 --                                     fieldAddr: Ptr;
41 612 --                                     fieldType: INTEGER)); OVERRIDE;
41 613 0- A  BEGIN
41 614 --      DoToField('TTEStyleCommand', NIL, bClass);
41 615 --      DoToField('fMode', @fMode, bInteger);
41 616 --      {$H-}
41 617 --      TextStyleFields('foldTextStyle', foldTextStyle, DoToField);
41 618 --      TextStyleFields('fNewTextStyle', fNewTextStyle, DoToField);
41 619 --      {$H+}
41 620 --      INHERITED Fields(DoToField);
41 621 0- A  END;
41 622 --      {$ENDC}
41 623 --
41 624 --      (*****
41 625 --      (*      T T E T y p i n g C o m m a n d
41 626 --      (*****
41 627 --
41 628 --      ($$ TERes)
41 629 --      {-----+
41 630 --      |      ITETypingCommand
41 631 --      +-----+
41 632 -- A  PROCEDURE TTETypingCommand.ITETypingCommand(itsTEView: TTEView; itsFirstChar: Char);
41 633 --      VAR
41 634 --          fi:          FailInfo;
41 635 --
41 636 --          {-----+
41 637 --          |      HdlInitFailed
41 638 --          +-----+
41 639 -- B  PROCEDURE HdlInitFailed(error: OSerr; message: LONGINT);
41 640 0- B  BEGIN
41 641 --          Free;
41 642 0- B  END;
41 643 --
41 644 0- A  BEGIN
41 645 --      ITECommand(itsTEView, cTyping, TRUE);
41 646 --
41 647 --      CatchFailures(fi, HdlInitFailed);
41 648 --
41 649 --      fNewStart := fHTE^.selStart;          { Start and end are the same }
41 650 --      fNewEnd := fNewStart;
41 651 --
41 652 --      fNewText := NewPermHandle(0);          { Allocate an empty block for text }
41 653 --      FailNIL(fNewText);
41 654 --
41 655 --      fCompleted := FALSE;                  { We've only just begun_ }
41 656 --      fFirstChar := itsFirstChar;          { Save character for DoIt }
41 657 --      Success(fi);
41 658 0- A  END;
41 659 --
41 660 --
41 661 --      ($$ TERes)
41 662 --      {-----+
41 663 --      |      Free
41 664 --      +-----+
41 665 -- A  PROCEDURE TTETypingCommand.Free; OVERRIDE;
41 666 0- A  BEGIN
41 667 --      IF fTEView.fTypingCommand = SELF THEN
41 668 --          fTEView.fTypingCommand := NIL;
41 669 --      INHERITED Free;
41 670 0- A  END;
41 671 --
41 672 --
41 673 --      ($$ TERes)
41 674 --      {-----+
41 675 --      |      DoNormalChar
41 676 --      +-----+
41 677 -- A  PROCEDURE TTETypingCommand.DoNormalChar(aChar: CHAR);
41 678 0- A  BEGIN
41 679 --      FailOSErr(PtrAndHand(Ptr(SUCC(ORD(@aChar))), { Append char to end of fNewText }
41 680 --                  fNewText, 1));
41 681 --      fNewEnd := SUCC(fNewEnd);              { Bump both end of "selection" }
41 682 --      fTextPad := SUCC(fTextPad);            { _and padding value }
41 683 --
41 684 --      SetHandleSize(fPadding,                { This SetHandleSize can't grow the handle, }
41 685 --                  MAX(-(fTextPad+fStylePad), 0)); { _so it shouldn't fail. }
41 686 0- A  END;
41 687 --
41 688 --      ($$ TERes)
41 689 --      {-----+
41 690 --      |      BkSpcLeft
41 691 --      +-----+
41 692 --      { User has backspaced to the left of the original starting point. First, copy the

```

```

41 693 --      character (which may be more than one byte long if we are using a non-Roman script)
41 694 --      to a temporary buffer. The assumption is that no character will ever be longer
41 695 --      than four bytes. Sorry, folks, MacApp does not support typing in any script with
41 696 --      more than 4 billion characters.
41 697 --      Next, copy the character to the front of foldText, and adjust foldStart, fNewStart,
41 698 --      and fNewEnd. Note that we do NOT check for MemSpacelsLow, since we want to let the
41 699 --      user delete characters. )
41 700 --
41 701 -- A  PROCEDURE TTETypingCommand.BkSpcLeft(theText: Handle; curStart: INTEGER);
41 702 --      TYPE
41 703 --          TSPtr = ^TextStyle;
41 704 --      VAR
41 705 --          savedSize:    INTEGER;
41 706 --          theHeight:    INTEGER;
41 707 --          theAscent:    INTEGER;
41 708 --          oldSize:      LONGINT;
41 709 --          whoCares:     LONGINT;
41 710 --          aTextStyle:   TSPtr;
41 711 --          savedChar:    PACKED ARRAY [0..3] OF CHAR;
41 712 --          delStyle:     TextStyle;
41 713 --      ($IFC qDebug)
41 714 --          savedPerm:    BOOLEAN;
41 715 --      ($ENDC)
41 716 --
41 717 0- A  BEGIN
41 718 --          savedSize := 1;
41 719 --          IF gConfiguration.hasScriptManager THEN
41 720 --              WHILE CharByte(theText^, curStart-savedSize) > 0 DO
41 721 --                  savedSize := SUCC(savedSize);
41 722 --                  curStart := curStart-savedSize;
41 723 --
41 724 --      ($IFC qDebug)
41 725 --          IF savedSize > 4 THEN
41 726 --              ProgramBreak('TTETypingCommand.AddCharacter: Character > 4 bytes');
41 727 --      ($ENDC)
41 728 --          IF savedSize = 1 THEN                                ( Slight speed optimization for normal case )
41 729 --      ($IFC qRangeCheck){$R-}{$$ENDC}
41 730 --              savedChar[0] := CharsHandle(theText)^[curStart]
41 731 --      ($IFC qRangeCheck){$R+}{$$ENDC}
41 732 --          ELSE
41 733 --              BlockMove(Ptr(ORD(theText^)+curStart), @savedChar, savedSize);
41 734 --
41 735 --          IF fTEView.fstyleType = kWithStyle THEN              ( Only do this if styles are around )
41 736 1-      BEGIN
41 737 --              TEGetStyle(curStart, delStyle,                    ( Get the style of the deleted character )
41 738 --              theHeight, theAscent, fHTE);                      ( (1 or 4 bytes, it's all only one style) )
41 739 --
41 740 --          IF NOT EqualBlocks(@delStyle,                          ( If style doesn't match first in the list )
41 741 --              @foldStyles^^.scrpStyleTab[0].scrpFont,
41 742 --              SIZEOF(TextStyle)) THEN
41 743 2-      BEGIN
41 744 --                  fTEView.fSpecsChanged := TRUE;                ( _then insert new style at head of list )
41 745 --                  ( User backspaced into new style! )
41 746 --
41 747 --                  oldSize :=                                     ( Make room for the new style element )
41 748 --                  GetHandleSize(Handle(foldStyles));
41 749 --                  SetHandleSize(Handle(foldStyles),
41 750 --                      oldSize + SIZEOF(ScrpSTElement));
41 751 --                  FailMemError;
41 752 --                  fStylePad := fStylePad + SIZEOF(ScrpSTElement);
41 753 --
41 754 --      ($H-)
41 755 --          WITH foldStyles^^.scrpStyleTab[0] DO
41 756 --              BlockMove(@scrpStartChar,                          ( Move entire array up one element's size )
41 757 --              Ptr(ORD(@scrpStartChar)+SIZEOF(ScrpSTElement)),
41 758 --              oldSize - SIZEOF(foldStyles^^.scrpNStyles));
41 759 --      ($H+)
41 760 --
41 761 --          foldStyles^^.scrpNStyles :=                             ( One more style )
41 762 --          SUCC(foldStyles^^.scrpNStyles);
41 763 3-      WITH foldStyles^^.scrpStyleTab[0] DO
41 764 --          BEGIN
41 765 --              scrpHeight := theHeight;                            ( Fill in the blanks )
41 766 --              scrpAscent := theAscent;
41 767 --              aTextStyle := TSPtr(@scrpFont);
41 768 --              aTextStyle^ := delStyle;
41 769 -2      END;
41 770 --
41 771 --          WITH foldStyles^^.scrpStyleTab[0] DO
41 772 --              scrpStartChar := PRED(scrpStartChar);              ( Regardless, back off offset by one )
41 773 -1      END;
41 774 --
41 775 --          SetHandleSize(fPadding, GetHandleSize(foldText) + savedSize + fStylePad);
41 776 --          FailMemError;
41 777 --          whoCares := Munger(foldText, 0, NIL, 0, @savedChar, savedSize);
41 778 --          foldStart := curStart;                                  ( Treat this as though original selection )
41 779 --          fNewStart := curStart;                                  ( _had included this character )
41 780 --          fNewEnd := curStart;
41 781 --          fTextPad := foldStart - foldEnd;
41 782 0- A  END;
41 783 --
41 784 --      ($S TERes)
41 785 --      [-----+
41 786 --      | BkSpcRight                                     | Handle backspace to the right of original selection
41 787 --      +-----+
41 788 -- A  PROCEDURE TTETypingCommand.BkSpcRight(theText: Handle; curStart: INTEGER);
41 789 --      VAR
41 790 --          savedSize:    INTEGER;
41 791 0- A  BEGIN

```

```

41 792 --         savedSize := 1;
41 793 --         IF gConfiguration.hasScriptManager THEN
41 794 --             WHILE CharByte(theText^, curStart - savedSize) > 0 DO
41 795 --                 savedSize := SUCC(savedSize);
41 796 --             SetHandleSize(fPadding, MAX(-(fTextPad - savedSize + fStylePad), 0));
41 797 --             FailMemError;
41 798 --             fNewEnd := fNewEnd - savedSize;
41 799 --             fTextPad := fTextPad - savedSize;
41 800 --
41 801 --             SetHandleSize(fNewText, fNewEnd-fNewStart); { Shouldn't fail as we're only shrinking it }
41 802 -- A   END;
41 803 --
41 804 --
41 805 --     {$S TERes}
41 806 --     {-----+
41 807 --     |   AddCharacter
41 808 --     +-----}
41 809 --     { ??? All this handle munging is expensive. Better would be to accumulate memory in
41 810 --       "chunks" of, say, 16 bytes so that this checking need not happen every time through.
41 811 --       Fortunately, the normal cases are not that bad. }
41 812 --
41 813 -- A   PROCEDURE TTTypingCommand.AddCharacter(aChar: CHAR);
41 814 --     VAR
41 815 --         theText:      Handle;
41 816 --         curSelStart:  INTEGER;
41 817 --         curSelEnd:    INTEGER;
41 818 --         savedPerm:    BOOLEAN;
41 819 --         fi:           FailInfo;
41 820 --
41 821 --     {-----+
41 822 --     |   HdlCharFailed
41 823 --     +-----}
41 824 -- B   PROCEDURE HdlCharFailed(error: OSErr; message: LONGINT);
41 825 -- B   BEGIN
41 826 --       savedPerm := PermAllocation(savedPerm);
41 827 -- B   END;
41 828 --
41 829 -- A   BEGIN {TTTypingCommand.AddCharacter}
41 830 --       WITH fHTE^ DO
41 831 --           BEGIN
41 832 --               curSelStart := selStart;
41 833 --               curSelEnd := selEnd;
41 834 --               theText := hText;
41 835 --           END;
41 836 --       CatchFailures(fi, HdlCharFailed);
41 837 --       savedPerm := PermAllocation(TRUE);
41 838 --
41 839 --       { Update the fNewText handle and other information. Note that because of backspace,
41 840 --         this can be tricky.}
41 841 --
41 842 --       IF aChar <> chBackspace THEN
41 843 --           DoNormalChar(aChar)
41 844 --       ELSE IF (curSelStart <= foldStart) AND
41 845 --           (curSelStart > 0) AND (curSelStart = curSelEnd) THEN
41 846 --           BkSpcLeft(theText, curSelStart)
41 847 --       ELSE IF fNewEnd > fNewStart THEN
41 848 --           BkSpcRight(theText, curSelStart);
41 849 --       savedPerm := PermAllocation(savedPerm);
41 850 --       Success(fi);
41 851 --
41 852 --       { Let TextEdit have the character, as either 1) we're adding a byte, so we know there
41 853 --         is a reserve tank, so the worst this will do is eat into it a little, or 2) we're
41 854 --         deleting a character, which can only decrease memory usage. }
41 855 --       TEKey(aChar, fHTE);
41 856 --
41 857 --       fTEView.SynchView(kRedraw);
41 858 --
41 859 --       { Now clean up the view.
41 860 --
41 861 --       ($IFC qDebug)
41 862 --       IF gTEIntenseDebugging THEN
41 863 --           BEGIN
41 864 --               WRITELN('foldText:');
41 865 --               WWAddText(foldText^, GetHandleSize(foldText)); WRITELN;
41 866 --               WRITELN('fNewText:');
41 867 --               WWAddText(fNewText^, GetHandleSize(fNewText)); WRITELN;
41 868 --               DumpTTECommand(SELF);
41 869 --           END;
41 870 --       ($ENDC)
41 871 --
41 872 --   END;
41 873 -- A   END;
41 874 --
41 875 --
41 876 --     {$S TERes}
41 877 --     {-----+
41 878 --     |   DoIt
41 879 --     +-----}
41 880 -- A   PROCEDURE TTTypingCommand.DoIt; OVERRIDE;
41 881 -- A   BEGIN
41 882 --       AddCharacter(fFirstChar);
41 883 --       ($IFC qDebug)
41 884 --       IF gTEIntenseDebugging THEN
41 885 --           DumpTTECommand(SELF);
41 886 --       ($ENDC)
41 887 -- A   END;
41 888 --
41 889 --
41 890 --     {$S TEdoCommand}

```



```

41 891 -- {-----+
41 892 -- | UndoIt
41 893 -- +-----}
41 894 -- A PROCEDURE TTTypingCommand.UndoIt; OVERRIDE;
41 895 0- A BEGIN
41 896 --     CompleteTyping;
41 897 --     INHERITED UndoIt;
41 898 --     {$IFC qDebug}
41 899 --         IF gTEIntenseDebugging THEN
41 900 --             DumpTTECommand(SELf);
41 901 --     {$ENDC}
41 902 -0 A END;
41 903 --
41 904 --
41 905 --     {$S TERes}
41 906 -- {-----+
41 907 -- | CompleteTyping | Mark as no more typing allowed & fix up styles
41 908 -- +-----}
41 909 -- A PROCEDURE TTTypingCommand.CompleteTyping;
41 910 -- VAR
41 911 --     i: INTEGER;
41 912 --     offset: LONGINT;
41 913 0- A BEGIN
41 914 --     fCompleted := TRUE;
41 915 --
41 916 --     IF fTEView.fStyleType = kWithStyle THEN
41 917 --         WITH fOldStyles DO
41 918 --             BEGIN
41 919 --                 offset := -scrpStyleTab[0].scrpStartChar;
41 920 --                 IF offset > 0 THEN
41 921 --                     FOR i := 0 TO scrpNStyles - 1 DO
41 922 --                         scrpStyleTab[i].scrpStartChar :=
41 923 --                             scrpStyleTab[i].scrpStartChar + offset;
41 924 --             END;
41 925 --     {$IFC qDebug}
41 926 --         IF gTEIntenseDebugging THEN
41 927 --             DumpTTECommand(SELf);
41 928 --     {$ENDC}
41 929 -0 A END;
41 930 --
41 931 --
41 932 --     {$IFC qDebug}
41 933 --     {$S TEFIELDS}
41 934 -- {-----+
41 935 -- | Fields |
41 936 -- +-----}
41 937 -- A PROCEDURE TTTypingCommand.Fields (PROCEDURE DoToField (fieldName: Str255;
41 938 --     fieldAddr: Ptr;
41 939 --     fieldType: INTEGER)); OVERRIDE;
41 940 0- A BEGIN
41 941 --     DoToField('TTTypingCommand', NIL, bClass);
41 942 --     DoToField('fCompleted', @fCompleted, bBoolean);
41 943 --     DoToField('fFirstChar', @fFirstChar, bBoolean);
41 944 --     INHERITED Fields(DoToField);
41 945 -0 A END;
41 946 -- {$ENDC}
38 626 --
38 627 -- END.

```

```

42 1 --      { $P }
42 2 --      { UViewCoords.p }
42 3 --      { Copyright © 1987-1988 by Apple Computer, Inc. All rights reserved. }
42 4 --
42 5 --      UNIT UViewCoords:
42 6 --
42 7 --      INTERFACE
42 8 --
42 9 --      {-----+
42 10 --      | Uses |
42 11 --      +-----+
42 12 --      USES
42 13 --      { $LOAD MacIntf.LOAD }
42 14 --      MemTypes, QuickDraw, OSIntf, ToolIntf, PackIntf,
42 15 --      { $LOAD }
42 16 --      UMAUtil;
42 17 --
42 18 --      {-----+
42 19 --      | Global constants |
42 20 --      +-----+
42 21 --      CONST
42 22 --      KMaxCoord      =      30000; { largest possible QuickDraw view coordinate
42 23 --                                   (QuickDraw maximum minus slop
42 24 --                                   for size of screen) }
42 25 --
42 26 --      {-----+
42 27 --      | Global routines |
42 28 --      +-----+
42 29 --      { Calculations with VPoints }
42 30 --
42 31 --      PROCEDURE PtToVPt (thePt: Point; VAR theVPt: VPoint);
42 32 --
42 33 --      FUNCTION VPtToPt (theVPt: VPoint): Point;
42 34 --
42 35 --      PROCEDURE AddVPt (srcVPt: VPoint; VAR dstVPt: VPoint);
42 36 --
42 37 --      PROCEDURE SubVPt (srcVPt: VPoint; VAR dstVPt: VPoint);
42 38 --
42 39 --      PROCEDURE SetVPt (VAR vPt: VPoint; h, v: VCoordinate);
42 40 --
42 41 --      FUNCTION EqualVPt (pt1, pt2: VPoint): BOOLEAN;
42 42 --
42 43 --
42 44 --      { Calculations with VRects }
42 45 --
42 46 --      PROCEDURE RectToVRect (theRect: Rect; VAR theVRect: VRect);
42 47 --
42 48 --      PROCEDURE VRectToRect (theVRect: VRect; VAR theRect: Rect);
42 49 --
42 50 --      PROCEDURE SetVRect (VAR r: VRect; left, top, right, bottom: VCoordinate);
42 51 --
42 52 --      PROCEDURE OffsetVRect (VAR r: VRect; dh, dv: VCoordinate);
42 53 --
42 54 --      PROCEDURE InsetVRect (VAR r: VRect; dh, dv: VCoordinate);
42 55 --
42 56 --      FUNCTION PtInVRect (pt: VPoint; r: VRect): BOOLEAN;
42 57 --
42 58 --      FUNCTION SectVRect (src1, src2: VRect; VAR dstRect: VRect): BOOLEAN;
42 59 --
42 60 --      FUNCTION EmptyVRect (r: VRect): BOOLEAN;
42 61 --
42 62 --      FUNCTION EqualVRect (rectA, rectB: VRect): BOOLEAN;
42 63 --
42 64 --      FUNCTION LengthVRect (r: VRect; vhs: VHSelect): VCoordinate;
42 65 --
42 66 --      PROCEDURE PinVRect (r: VRect; VAR pt: VPoint);
42 67 --
42 68 --      PROCEDURE UnionVRect (src1, src2: VRect; VAR dstRect: VRect);
42 69 --
42 70 --      PROCEDURE Pt2VRect (topLeft, botRight: VPoint; VAR dstRect: VRect);
42 71 --
42 72 --      IMPLEMENTATION
42 73 --
42 74 --      { $I UViewCoords.incl.p }
42 75 --
42 76 --      END { UViewCoords }.

```

1. UAssociation.p  
 2. UAssociation.incl.p  
 3. UBusyCursor.p  
 4. UBusyCursor.incl.p  
 5. UDialog.p  
 6. UDialog.incl.p  
 7. UFailure.p  
 8. UFailure.incl.p  
 9. UGridView.p  
 10. UGridView.incl.p  
 11. UList.p  
 12. UList.incl.p  
 13. UMacApp.p  
 14. UMacApp.Globals.p  
 15. UMacApp.TEvtHandler.p  
 16. UMacApp.TApplication.p  
 17. UMacApp.TDocument.p  
 18. UMacApp.TView.p  
 19. UMacApp.TWindow.p  
 20. UMacApp.TScroller.p  
 21. UMacApp.TControls.p  
 22. UMacApp.TCommand.p  
 23. UMacApp.TDeskScrapView.p  
 24. UMacApp.TPrintHandler.p  
 25. UMAUtil.p  
 26. UMAUtil.incl.p  
 27. UMemory.p  
 28. UMemory.incl.p  
 29. UMenuSetup.p  
 30. UMenuSetup.incl.p  
 31. UObject.p  
 32. UObject.incl.p  
 33. ObjLib.incl.p  
 34. UPatch.p  
 35. UPatch.incl.p  
 36. UPrinting.p  
 37. UPrinting.incl.p  
 38. UTEView.p  
 39. UTEView.Globals.p  
 40. UTEView.TTEView.p  
 41. UTEView.TTECommand.p  
 42. UViewCoords.p

-\*-

|                 |          |          |           |          |          |          |          |  |
|-----------------|----------|----------|-----------|----------|----------|----------|----------|--|
| %_GetA5         | 609 (12) | 212*(25) | 1128 (28) | 77 (33)  | 225 (32) |          |          |  |
| %_GetA6         | 47 (8)   | 98 (8)   | 126 (8)   | 213*(25) | 76 (33)  | 176 (33) | 179 (33) |  |
| %_GetA7         | 214*(25) |          |           |          |          |          |          |  |
| %_InitObj       | 316*(14) | 353 (14) | 15*(33)   | 52 (33)  |          |          |          |  |
| %_InObj         | 24*(12)  | 123 (12) | 440 (12)  | 11*(33)  | 139 (33) | 226 (32) |          |  |
| %_METHCACHE     | 34*(33)  |          |           |          |          |          |          |  |
| %_MethCache     | 51-(33)  |          |           |          |          |          |          |  |
| %_OBCHK         | 147*(31) | 135*(33) | 137-(33)  | 142 (33) |          |          |          |  |
| %_OBDISP        | 156*(31) | 123*(33) |           |          |          |          |          |  |
| %_OBNEW         | 153*(31) | 61*(33)  | 83 (33)   |          |          |          |          |  |
| %_Pgml          | 143*(31) | 37*(33)  |           |          |          |          |          |  |
| %_RTS1          | 14*(33)  |          |           |          |          |          |          |  |
| %_SetClassIndex | 12*(33)  | 108 (33) |           |          |          |          |          |  |

-A-

|                      |           |           |           |           |           |           |           |           |
|----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| a                    | 297*(25)  | 300*(25)  | 500*(26)  | 502 (26)  | 505 (26)  | 514*(26)  | 516 (26)  | 517 (26)  |
| AbandonUndoClipboard | 1002*(13) | 173*(16)  | 470 (16)  | 562 (16)  |           |           |           |           |
| AboutToLoseControl   | 1006*(13) | 190*(16)  | 1678 (16) | 2417 (16) | 2713 (16) |           |           |           |
| AboutToSave          | 1231*(13) | 135*(17)  | 1103 (17) |           |           |           |           |           |
| ABS                  | 1188 (10) | 1272 (10) | 2104 (10) | 2127 (10) | 267 (15)  | 268 (15)  | 596 (17)  | 502 (20)  |
|                      | 502 (20)  | 785 (21)  | 785 (21)  | 487 (37)  | 488 (37)  |           |           |           |
| Abs                  | 3284 (16) | 3285 (16) |           |           |           |           |           |           |
| AbsorbScrapStuff     | 1012*(13) | 230 (16)  | 239*(16)  | 380 (16)  | 395 (16)  | 1894 (16) |           |           |
| aButton              | 17*(6)    | 34 (6)    | 34 (6)    | 34 (6)    |           |           |           |           |
| aCell                | 240*(9)   | 244*(9)   | 255*(9)   | 262*(9)   | 265*(9)   | 295*(9)   | 299*(9)   | 303*(9)   |
|                      | 343*(9)   | 350*(9)   | 356*(9)   | 469*(9)   | 479*(9)   | 565*(9)   | 588*(9)   | 678*(9)   |
|                      | 682*(9)   | 309*(10)  | 367*(10)  | 381*(10)  | 395*(10)  | 400 (10)  | 400 (10)  | 401 (10)  |
|                      | 401 (10)  | 405 (10)  | 406 (10)  | 414 (10)  | 415 (10)  | 494*(10)  | 496 (10)  | 496 (10)  |
|                      | 497 (10)  | 497 (10)  | 499 (10)  | 629*(10)  | 634 (10)  | 635 (10)  | 642 (10)  | 772*(10)  |
|                      | 804 (10)  | 805 (10)  | 806 (10)  | 807 (10)  | 823*(10)  | 1082*(10) | 1089*(10) | 1112 (10) |
|                      | 1113 (10) | 1114 (10) | 1125*(10) | 1139*(10) | 1143*(10) | 1145 (10) | 1147 (10) | 1404*(10) |
|                      | 1409 (10) | 1453*(10) | 1457-(10) | 1458 (10) | 1460 (10) | 1461 (10) | 1463 (10) | 1463 (10) |
|                      | 1464 (10) | 1464 (10) | 1762*(10) | 1764 (10) | 1841*(10) | 1843 (10) | 2075*(10) | 2087 (10) |
|                      | 2089 (10) | 2094 (10) | 2100 (10) | 2100 (10) | 2104 (10) | 2104 (10) | 2110 (10) | 2112 (10) |
|                      | 2117 (10) | 2123 (10) | 2123 (10) | 2127 (10) | 2127 (10) | 2129 (10) | 2315*(10) | 2319 (10) |
|                      | 2321 (10) | 2389*(10) | 2496*(10) | 2498 (10) | 2574*(10) | 2576 (10) | 2627*(10) | 2629 (10) |
|                      | 2679*(10) | 2682 (10) | 2683 (10) | 2684 (10) | 2723*(10) | 2725 (10) | 2726 (10) | 2728 (10) |
|                      | 2854*(10) | 2870*(10) | 2890 (10) | 2891 (10) | 2892 (10) | 2893 (10) | 2897 (10) | 2898 (10) |
|                      | 2899 (10) | 2900 (10) | 2933*(10) | 2941-(10) | 2943 (10) | 2944 (10) | 2944 (10) | 2945 (10) |
|                      | 2945 (10) | 2946 (10) | 2949 (10) | 2954 (10) | 2955 (10) | 2956 (10) | 2963 (10) | 2975 (10) |
|                      | 2981 (10) | 2993 (10) | 3007 (10) | 3018 (10) |           |           |           |           |
| aCellSelectCommand   | 625*(10)  | 640 (10)  | 641 (10)  | 642 (10)  | 644 (10)  |           |           |           |
| aChar                | 562*(38)  | 571*(38)  | 677*(41)  | 679 (41)  | 813*(41)  | 842 (41)  | 843 (41)  | 858 (41)  |
| aCheckBox            | 18*(6)    | 35 (6)    | 35 (6)    | 35 (6)    |           |           |           |           |
| aChunkNum            | 306*(9)   | 309*(9)   | 1163*(10) | 1173-(10) | 1181-(10) | 1192-(10) | 1198-(10) | 1199 (10) |
|                      | 1201 (10) | 1206-(10) | 1213 (10) | 1215 (10) | 1216 (10) | 1218-(10) | 1218 (10) | 1224-(10) |
|                      | 1224 (10) | 1225 (10) | 1226 (10) | 1235 (10) | 1247*(10) | 1257-(10) | 1265-(10) | 1276-(10) |
|                      | 1282-(10) | 1283 (10) | 1285 (10) | 1290-(10) | 1297 (10) | 1299 (10) | 1300 (10) | 1302-(10) |
|                      | 1302 (10) | 1308-(10) | 1308 (10) | 1309 (10) | 1310 (10) | 1319 (10) |           |           |
| aClearCommand        | 761*(40)  | 781 (40)  | 782 (40)  | 783 (40)  | 784 (40)  |           |           |           |
| aClipType            | 2891*(13) | 73*(14)   | 76 (14)   | 79 (14)   |           |           |           |           |
| aCluster             | 20*(6)    | 37 (6)    | 37 (6)    | 37 (6)    |           |           |           |           |
| aCmd                 | 193*(29)  | 207*(29)  | 215*(29)  | 218*(29)  | 243*(29)  | 248*(29)  | 253*(29)  | 257*(29)  |
|                      | 265*(29)  | 100*(30)  | 108 (30)  | 110 (30)  | 111 (30)  | 124 (30)  | 130 (30)  | 148*(30)  |

|                      |            |            |            |            |            |            |            |            |
|----------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                      | 156 (30)   | 243* (30)  | 250 (30)   | 252 (30)   | 268* (30)  | 275 (30)   | 277 (30)   | 517* (30)  |
|                      | 524 (30)   | 526 (30)   | 539* (30)  | 546 (30)   | 548 (30)   | 561* (30)  | 566 (30)   | 576* (30)  |
| aCmdNumber           | 586 (30)   | 596* (30)  | 603 (30)   | 605 (30)   |            |            |            |            |
|                      | 586* (5)   | 3039* (6)  | 3047 (6)   | 3053 (6)   | 642* (13)  | 952* (13)  | 994* (13)  | 1404* (13) |
|                      | 1702* (13) | 245* (15)  | 248 (15)   | 252 (15)   | 701* (16)  | 721 (16)   | 731 (16)   | 735 (16)   |
|                      | 736 (16)   | 831 (16)   | 2801* (16) | 2809 (16)  | 2811 (16)  | 2821 (16)  | 2821 (16)  | 2824 (16)  |
|                      | 2831 (16)  | 299* (17)  | 316 (17)   | 320 (17)   | 324 (17)   | 325 (17)   | 326 (17)   | 343 (17)   |
|                      | 784* (18)  | 786 (18)   | 790 (18)   | 792 (18)   | 794 (18)   | 401* (36)  | 654* (37)  | 660 (37)   |
|                      | 666 (37)   | 672 (37)   | 684 (37)   | 182* (38)  | 192* (38)  | 757* (40)  | 764 (40)   | 769 (40)   |
| aCmGrControl         | 783 (40)   | 828* (40)  | 838 (40)   | 842 (40)   | 857 (40)   | 873 (40)   |            |            |
| aCohandler           | 12* (21)   | 20 (21)    | 628* (21)  | 641* (21)  | 644 (21)   | 649 (21)   | 651 (21)   | 652 (21)   |
| aCol                 | 794* (13)  | 1834* (16) | 1841 (16)  | 1842 (16)  | 1850 (16)  | 1853 (16)  | 1855 (16)  |            |
|                      | 235* (9)   | 269* (9)   | 279* (9)   | 306* (9)   | 315* (9)   | 320* (9)   | 325* (9)   | 362* (9)   |
|                      | 329* (10)  | 432* (10)  | 441 (10)   | 441 (10)   | 445 (10)   | 458 (10)   | 463 (10)   | 466 (10)   |
|                      | 466 (10)   | 624* (10)  | 633 (10)   | 634 (10)   | 660 (10)   | 669 (10)   | 839* (10)  | 853 (10)   |
|                      | 853 (10)   | 859 (10)   | 869 (10)   | 871 (10)   | 876 (10)   | 876 (10)   | 877 (10)   | 1163* (10) |
|                      | 1171 (10)  | 1171 (10)  | 1177* (10) | 1179 (10)  | 1188 (10)  | 1190 (10)  | 1190 (10)  | 1196 (10)  |
|                      | 1211 (10)  | 1213 (10)  | 1228 (10)  | 1230 (10)  | 1234 (10)  | 1342* (10) | 1350 (10)  | 1350 (10)  |
|                      | 1353 (10)  | 1361 (10)  | 1449* (10) | 1461* (10) | 1482* (10) | 1482 (10)  | 1515* (10) | 1527 (10)  |
|                      | 1533 (10)  | 1542 (10)  | 1550 (10)  | 1562 (10)  | 1564 (10)  | 1603 (10)  | 1603 (10)  | 1882* (10) |
|                      | 1885 (10)  | 1885 (10)  | 1888 (10)  | 1896 (10)  | 1897 (10)  |            |            |            |
| aColor               | 69* (6)    | 79 (6)     | 93* (6)    | 97 (6)     | 99 (6)     | 101 (6)    | 109* (6)   | 117 (6)    |
|                      | 121 (6)    | 123 (6)    | 125 (6)    | 261* (25)  | 335* (25)  | 290* (26)  | 300 (26)   | 314* (26)  |
|                      | 318 (26)   | 320 (26)   | 322 (26)   | 780* (26)  | 788 (26)   | 792 (26)   | 794 (26)   | 796 (26)   |
| aColumn              | 738* (9)   | 761* (9)   | 3113* (10) | 3116 (10)  | 3147* (10) | 3151 (10)  |            |            |
| aColumnSelectCommand | 627* (10)  | 658 (10)   | 659 (10)   | 660 (10)   | 662 (10)   |            |            |            |
| aCommand             | 3021* (6)  | 3026* (6)  | 3027 (6)   | 3028 (6)   | 3029 (6)   | 3042* (6)  | 3047* (6)  | 3048 (6)   |
|                      | 3049 (6)   | 3050 (6)   | 3065* (6)  | 3074* (6)  | 3075 (6)   | 3076 (6)   | 3077 (6)   | 1331* (16) |
|                      | 1339 (16)  | 1343* (16) | 1345 (16)  | 1555* (16) | 1615 (16)  | 1616 (16)  | 1617 (16)  | 788* (17)  |
|                      | 790* (17)  |            |            |            |            |            |            |            |
| aControl             | 15* (6)    | 32 (6)     | 32 (6)     | 32 (6)     |            |            |            |            |
| aControlTracker      | 305* (21)  | 307 (21)   | 308 (21)   | 309 (21)   | 310 (21)   |            |            |            |
| Activate             | 1542* (13) | 1957* (13) | 2087* (13) | 2302* (13) | 1318 (16)  | 1690 (16)  | 306* (18)  | 313 (18)   |
|                      | 253* (19)  | 265 (19)   | 271 (19)   | 385 (19)   | 216* (21)  | 1319* (21) |            |            |
| activateEvt          | 642 (16)   | 1409 (16)  | 2557 (16)  | 2594 (16)  | 91 (37)    |            |            |            |
| ActivateSubview      | 311* (18)  | 328 (18)   |            |            |            |            |            |            |
| active               | 994 (40)   |            |            |            |            |            |            |            |
| activeFlag           | 2558 (16)  |            |            |            |            |            |            |            |
| activMask            | 702 (19)   |            |            |            |            |            |            |            |
| aCtlMgr              | 16* (6)    | 33 (6)     | 33 (6)     | 33 (6)     |            |            |            |            |
| aCutCopyCommand      | 759* (40)  | 767 (40)   | 768 (40)   | 769 (40)   | 770 (40)   | 831* (40)  |            |            |
| adapted              | 1887* (13) |            |            |            |            |            |            |            |
| AdaptToScreen        | 2008* (13) | 152 (19)   | 298* (19)  |            |            |            |            |            |
| aDataHandle          | 1027* (13) | 1665* (13) | 2895* (13) | 1117* (14) | 1120 (14)  | 1121 (14)  | 1122 (14)  | 1122 (14)  |
|                      | 1123 (14)  | 1090* (16) | 1100 (16)  | 1237* (18) | 1248 (18)  | 259* (38)  | 1076* (40) | 1114 (40)  |
|                      | 1117 (40)  | 1122 (40)  | 1144 (40)  | 1151 (40)  | 1160 (40)  |            |            |            |
| AddAllRsrc           | 269* (27)  | 101* (28)  | 473 (28)   | 474 (28)   | 475 (28)   | 476 (28)   | 477 (28)   |            |
| AddCharacter         | 571* (38)  | 747 (40)   | 813* (41)  | 882 (41)   |            |            |            |            |
| AddDocument          | 894* (13)  | 252* (16)  | 2169 (16)  | 2228 (16)  |            |            |            |            |
| AddFreeWindow        | 897* (13)  | 261* (16)  | 737 (19)   |            |            |            |            |            |
| AddHandle            | 264* (27)  | 124 (28)   | 136* (28)  |            |            |            |            |            |
| addIt                | 795* (13)  | 1834* (16) | 1839 (16)  |            |            |            |            |            |
| AddObjectToInspector | 61* (14)   | 85 (16)    | 86 (16)    | 87 (16)    | 19* (33)   | 111 (33)   | 212 (33)   |            |
| AddPt                | 322 (19)   | 958 (19)   | 239 (21)   | 1069 (37)  | 1070 (37)  | 1098 (37)  |            |            |
| addr                 | 44* (38)   |            |            |            |            |            |            |            |
| AddResMenu           | 126 (16)   |            |            |            |            |            |            |            |
| address              | 31* (18)   | 34 (18)    |            |            |            |            |            |            |
| AddSegSizes          | 357* (27)  | 148* (28)  | 182* (28)  | 487 (28)   |            |            |            |            |
| addSize              | 1399 (40)  |            |            |            |            |            |            |            |
| AddSubview           | 2586 (6)   | 1789* (13) | 856 (14)   | 869 (14)   | 872 (14)   | 944 (14)   | 948 (14)   | 2765 (16)  |
|                      | 208 (18)   | 335* (18)  | 137* (20)  | 140 (20)   |            |            |            |            |
| AddSubview           | 1528* (13) |            |            |            |            |            |            |            |
| AddToHLRegion        | 494* (10)  | 532 (10)   |            |            |            |            |            |            |
| AddView              | 1409* (13) | 125* (17)  | 210 (18)   |            |            |            |            |            |
| AddVpt               | 3135 (16)  | 3136 (16)  | 3304 (16)  | 561 (18)   | 1404 (18)  | 1420 (18)  | 1519 (18)  | 278 (20)   |
|                      | 367 (20)   | 664 (20)   | 35* (42)   |            |            |            |            |            |
| AddWindow            | 1412* (13) | 146* (17)  | 729 (19)   |            |            |            |            |            |
| aDeskScrapView       | 477* (14)  | 696 (14)   | 696 (14)   | 696 (14)   |            |            |            |            |
| aDialog              | 58* (37)   | 77 (37)    | 78 (37)    |            |            |            |            |            |
| aDialogTEView        | 259* (6)   | 286* (6)   | 287 (6)    | 302* (6)   | 331* (6)   | 332 (6)    | 614* (6)   | 617 (6)    |
|                      | 618 (6)    | 619 (6)    | 621 (6)    | 623 (6)    | 624 (6)    | 625 (6)    |            |            |
| aDialogView          | 14* (6)    | 31 (6)     | 31 (6)     | 31 (6)     | 1254* (6)  | 1280* (6)  | 1281 (6)   | 1282 (6)   |
|                      | 2783* (6)  | 2792* (6)  | 2793 (6)   | 2794 (6)   |            |            |            |            |
| AdjustBotRight       | 463* (5)   | 2187* (6)  | 2517 (6)   |            |            |            |            |            |
| AdjustScrollBars     | 1815* (13) | 148* (20)  | 379 (20)   | 416 (20)   |            |            |            |            |
| AdjustSize           | 923 (10)   | 1020 (10)  | 1601 (10)  | 1701 (10)  | 1560* (13) | 376* (18)  | 1674 (18)  | 217 (23)   |
|                      | 300 (23)   | 1663 (37)  | 1583 (40)  |            |            |            |            |            |
| adnDummy             | 149* (25)  |            |            |            |            |            |            |            |
| adnFrame             | 862* (26)  | 1069 (26)  | 1070 (26)  |            |            |            |            |            |
| adnLineBottom        | 33 (5)     | 2361 (6)   | 396 (18)   | 510 (18)   | 147* (25)  | 862 (26)   | 1075 (26)  |            |
| adnLineLeft          | 33 (5)     | 2361 (6)   | 396 (18)   | 508 (18)   | 146* (25)  | 862 (26)   | 1074 (26)  |            |
| adnLineRight         | 33 (5)     | 2361 (6)   | 396 (18)   | 512 (18)   | 148* (25)  | 862 (26)   | 1076 (26)  |            |
| adnLineTop           | 33 (5)     | 2361 (6)   | 396 (18)   | 506 (18)   | 145* (25)  | 862 (26)   | 1073 (26)  |            |
| adnOval              | 487 (18)   | 150* (25)  | 1079 (26)  |            |            |            |            |            |
| adnRRect             | 490 (18)   | 151* (25)  | 1080 (26)  |            |            |            |            |            |
| adnShadow            | 2361 (6)   | 483 (18)   | 266 (21)   | 152* (25)  | 1081 (26)  |            |            |            |
| aDocument            | 907* (13)  | 514* (16)  | 517 (16)   | 704* (16)  | 1051* (16) | 1064* (16) | 1066 (16)  | 2132* (16) |
|                      | 2142 (16)  | 2147* (16) | 2150* (16) | 2151 (16)  | 2152 (16)  | 2153 (16)  | 2155 (16)  | 2158 (16)  |
|                      | 2159 (16)  | 2161 (16)  | 2164 (16)  | 2169 (16)  | 2174 (16)  | 2186* (16) | 2197 (16)  | 2207* (16) |
|                      | 2217 (16)  | 2219* (16) | 2221 (16)  | 2222 (16)  | 2223 (16)  | 2228 (16)  | 2235 (16)  | 2432* (16) |
|                      | 2442 (16)  | 2446* (16) | 2449* (16) | 2450 (16)  | 2451 (16)  | 2456 (16)  | 2477 (16)  |            |
| Adorn                | 3123 (6)   | 1596* (13) | 394* (18)  | 325 (21)   | 713 (21)   |            |            |            |
| AdornCol             | 235* (9)   | 329* (10)  | 735 (10)   |            |            |            |            |            |
| adornCols            | 100* (9)   | 180* (9)   | 435* (9)   | 528* (9)   | 38* (10)   | 54 (10)    | 154 (10)   | 266* (10)  |
|                      | 2207* (10) | 2218 (10)  | 2447* (10) | 2458 (10)  |            |            |            |            |
| AdornPage            | 294* (36)  | 348* (37)  | 1606 (37)  |            |            |            |            |            |
| adornPieces          | 145* (25)  | 154 (25)   |            |            |            |            |            |            |
| AdornRow             | 236* (9)   | 339* (10)  | 754 (10)   |            |            |            |            |            |
| adornRows            | 99* (9)    | 179* (9)   | 434* (9)   | 527* (9)   | 37* (10)   | 53 (10)    | 153 (10)   | 265* (10)  |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| aFirstHandler     | 2206*(10) | 2218 (10) | 2446*(10) | 2458 (10) |           |           |           |           |  |
| aFontInfo         | 799*(13)  | 804*(13)  | 881*(16)  | 891 (16)  | 1021*(16) | 1032 (16) |           |           |  |
| aHandle           | 1555*(40) | 1559 (40) | 1561 (40) |           |           |           |           |           |  |
|                   | 110*(15)  | 129=(15)  | 130 (15)  | 131 (15)  | 132 (15)  | 136 (15)  | 137 (15)  | 190 (15)  |  |
|                   | 44*(18)   | 48=(18)   | 49 (18)   | 50 (18)   | 52 (18)   | 57 (18)   | 157*(23)  | 167 (23)  |  |
|                   | 168 (23)  | 206=(23)  | 207 (23)  | 211 (23)  | 216 (23)  | 237*(25)  | 156*(26)  | 159 (26)  |  |
|                   | 160 (26)  | 1081*(40) | 1091 (40) | 1093 (40) | 1094 (40) | 1108=(40) | 1138=(40) | 1141 (40) |  |
|                   | 1143 (40) | 1147 (40) | 1148 (40) | 1150 (40) | 1152 (40) | 1157 (40) | 1158=(40) | 1600*(40) |  |
|                   | 1608=(40) | 1609 (40) | 1610 (40) |           |           |           |           |           |  |
| AHandleList       | 153 (27)  | 154*(27)  |           |           |           |           |           |           |  |
| aHeight           | 326*( 9)  | 332*( 9)  | 338*( 9)  | 363*( 9)  | 612*( 9)  | 1615*(10) | 1643 (10) | 1651 (10) |  |
|                   | 1676 (10) | 1693 (10) | 1700 (10) | 1727*(10) | 1729 (10) | 1751*(10) | 1753 (10) | 1908*(10) |  |
|                   | 1920 (10) | 1923 (10) | 2738*(10) | 2740 (10) |           |           |           |           |  |
| aHRunArray        | 45*(10)   | 73=(10)   | 74 (10)   | 75 (10)   | 77=(10)   | 78 (10)   | 79 (10)   | 142*(10)  |  |
|                   | 179=(10)  | 180 (10)  | 181 (10)  | 183=(10)  | 184 (10)  | 185 (10)  |           |           |  |
| aIcon             | 21*( 6)   | 38 ( 6)   | 38 ( 6)   | 38 ( 6)   |           |           |           |           |  |
| aKeyCode          | 95*( 5)   | 584*( 5)  | 493*( 6)  | 502 ( 6)  | 3018*( 6) | 3026 ( 6) | 3032 ( 6) | 633*(13)  |  |
|                   | 233*(15)  | 236 (15)  | 178*(38)  | 695*(40)  |           |           |           |           |  |
| aLabel            | 470*(25)  | 473*(25)  | 476*(25)  | 479*(25)  | 482*(25)  | 485*(25)  | 1182*(26) | 1185 (26) |  |
|                   | 1208*(26) | 1211 (26) | 1232*(26) | 1235 (26) | 1257*(26) | 1260 (26) | 1280*(26) | 1283 (26) |  |
|                   | 1304*(26) | 1307 (26) |           |           |           |           |           |           |  |
| Alert             | 804 (14)  | 1475 (37) |           |           |           |           |           |           |  |
| alertID           | 2918*(13) | 170*(14)  | 181=(14)  | 188=(14)  | 193=(14)  | 220=(14)  | 223 (14)  | 797*(14)  |  |
|                   | 804 (14)  |           |           |           |           |           |           |           |  |
| alertId           | 2921*(13) | 1232*(14) | 1235 (14) |           |           |           |           |           |  |
| alias             | 902*(26)  | 913 (26)  | 927 (26)  | 935=(26)  | 937 (26)  | 1069 (26) |           |           |  |
| AllCellsDo        | 262*( 9)  | 309*(10)  |           |           |           |           |           |           |  |
| AllItemsDo        | 562*( 9)  | 2485*(10) |           |           |           |           |           |           |  |
| AllocBlock        | 63*(35)   | 88=(35)   | 109 (35)  | 130 (35)  | 151 (35)  | 185 (35)  |           |           |  |
| allocLim          | 55*(28)   | 223*(28)  | 248 (28)  | 257 (28)  |           |           |           |           |  |
| allowIt           | 541*(36)  | 137*(37)  | 143 (37)  |           |           |           |           |           |  |
| AllowsMenuAccess  | 1994*(13) | 1826 (16) | 334*(19)  | 338=(19)  |           |           |           |           |  |
| ALoadMacAppSeq    | 61*(28)   | 516 (28)  |           |           |           |           |           |           |  |
| alreadyDidCommand | 1952*(16) | 1975=(16) | 1994=(16) | 2000=(16) | 2003 (16) |           |           |           |  |
| AlreadyOpen       | 828*(13)  | 270*(16)  | 313=(16)  | 2215 (16) | 938 (17)  |           |           |           |  |
| alter             | 303*(27)  | 325*(28)  | 334 (28)  |           |           |           |           |           |  |
| aMenu             | 2030*( 6) | 2051=( 6) | 2052 ( 6) | 2053 ( 6) | 2054 ( 6) | 2084*( 6) | 2107=( 6) | 2108 ( 6) |  |
|                   | 2109 ( 6) | 2110 ( 6) | 2488*( 6) | 70*(16)   | 119=(16)  | 120 (16)  | 121 (16)  | 124=(16)  |  |
|                   | 125 (16)  | 126 (16)  | 138=(16)  | 139 (16)  | 140 (16)  | 150*(30)  | 156 (30)  | 157 (30)  |  |
| aMenuHandle       | 2245*( 6) | 2268=( 6) | 210*(29)  | 236*(29)  | 32*(30)   | 166*(30)  | 246*(30)  | 253=(30)  |  |
|                   | 254 (30)  | 256 (30)  | 258 (30)  | 271*(30)  | 278=(30)  | 279 (30)  | 282 (30)  | 284 (30)  |  |
|                   | 285 (30)  | 390*(30)  | 392 (30)  | 405*(30)  | 408 (30)  | 424*(30)  | 432*(30)  | 437 (30)  |  |
|                   | 451 (30)  | 455 (30)  | 457 (30)  | 460 (30)  | 467*(30)  | 472 (30)  | 493 (30)  | 520*(30)  |  |
|                   | 527=(30)  | 528 (30)  | 529 (30)  | 542*(30)  | 549=(30)  | 550 (30)  | 551 (30)  | 599*(30)  |  |
|                   | 606=(30)  | 607 (30)  | 608 (30)  |           |           |           |           |           |  |
| AMenuList         | 180 (30)  | 181*(30)  |           |           |           |           |           |           |  |
| aModulus          | 323*(25)  | 738*(26)  | 741 (26)  | 741 (26)  | 741 (26)  |           |           |           |  |
| amplitude         | 238=( 8)  | 242=( 8)  |           |           |           |           |           |           |  |
| amtMoved          | 3111*(16) | 3282=(16) | 3283 (16) | 3284 (16) | 3285 (16) |           |           |           |  |
| amtPerPage        | 2367*(13) | 68*(24)   | 214*(36)  | 479*(37)  | 485 (37)  |           |           |           |  |
| anAppFile         | 817*(13)  | 834*(13)  | 860*(13)  | 865*(13)  | 1213*(13) | 329*(16)  | 356 (16)  | 360 (16)  |  |
|                   | 361 (16)  | 408*(16)  | 457 (16)  | 458 (16)  | 459 (16)  | 460 (16)  | 707*(16)  | 735 (16)  |  |
|                   | 736 (16)  | 1435*(16) | 1451 (16) | 1496 (16) | 1498 (16) | 1505 (16) | 1508 (16) | 2183*(16) |  |
|                   | 2200 (16) | 2215 (16) | 2215 (16) | 2219 (16) | 2221 (16) | 2430*(16) | 2449 (16) | 2450 (16) |  |
|                   | 799*(17)  | 823 (17)  | 826 (17)  | 827 (17)  | 831 (17)  | 832 (17)  | 848 (17)  | 848 (17)  |  |
|                   | 893 (17)  | 893 (17)  | 958*(17)  | 995 (17)  | 996 (17)  |           |           |           |  |
| anAssociation     | 258*( 6)  | 276 ( 6)  | 277 ( 6)  | 278 ( 6)  | 279 ( 6)  | 301*( 6)  | 323 ( 6)  | 324 ( 6)  |  |
|                   | 325 ( 6)  | 326 ( 6)  |           |           |           |           |           |           |  |
| anchorCell        | 682*( 9)  | 2870*(10) | 2897 (10) | 2898 (10) | 2899 (10) | 2900 (10) | 2934*(10) | 2992=(10) |  |
|                   | 2993 (10) |           |           |           |           |           |           |           |  |
| anchorPoint       | 653*( 9)  | 686*( 9)  | 691*( 9)  | 2820*(10) | 2930*(10) | 2992 (10) | 3029*(10) | 2122*(13) |  |
|                   | 2128*(13) | 2132*(13) | 2480*(13) | 2486*(13) | 2493*(13) | 2535*(13) | 2538*(13) | 2541*(13) |  |
|                   | 3107*(16) | 3195 (16) | 3209 (16) | 3231 (16) | 3256=(16) | 3261=(16) | 3283 (16) | 54*(21)   |  |
|                   | 58 (21)   | 65*(21)   | 68 (21)   | 76*(21)   | 80 (21)   | 451*(21)  | 460*(21)  | 470*(21)  |  |
|                   | 71*(22)   | 81*(22)   | 87 (22)   | 98*(22)   |           |           |           |           |  |
| anEditText        | 25*( 6)   | 42 ( 6)   | 42 ( 6)   | 42 ( 6)   |           |           |           |           |  |
| anEntriesList     | 94*( 2)   | 97 ( 2)   | 98 ( 2)   | 99 ( 2)   | 100 ( 2)  |           |           |           |  |
| anEntry           | 193*( 2)  | 203 ( 2)  | 216=( 2)  | 217 ( 2)  | 220 ( 2)  | 226 ( 2)  | 227 ( 2)  | 228 ( 2)  |  |
|                   | 229 ( 2)  | 1157*(14) | 1160=(14) | 1161 (14) | 59*(15)   | 73=(15)   | 74 (15)   | 75 (15)   |  |
| anEvent           | 704*(13)  | 1120*(16) | 1153 (16) | 1160 (16) | 1166 (16) | 1186 (16) | 1188 (16) | 1192 (16) |  |
|                   | 1193 (16) | 1664*(16) | 3364*(16) | 3367 (16) | 3368 (16) | 691*(19)  | 702 (19)  | 703 (19)  |  |
|                   | 59*(37)   | 67 (37)   | 75 (37)   | 77 (37)   | 91 (37)   | 91 (37)   | 97 (37)   |           |  |
| anEvtHandler      | 800*(13)  | 805*(13)  | 506*(16)  | 508 (16)  | 882*(16)  | 1022*(16) | 1332*(16) | 1337*(16) |  |
|                   | 1339 (16) | 1344=(16) | 1754*(16) | 1760 (16) | 1765 (16) |           |           |           |  |
| aNewDocument      | 894*(13)  | 252*(16)  | 254 (16)  |           |           |           |           |           |  |
| anHPrint          | 228*(37)  | 234 (37)  | 1694*(37) | 1704 (37) | 1716=(37) | 1718 (37) | 1725 (37) | 1743 (37) |  |
|                   | 1794*(37) | 1797=(37) | 1798 (37) | 1799 (37) |           |           |           |           |  |
| anHTE             | 1204*(40) | 1225=(40) | 1227=(40) | 1232 (40) | 1233 (40) |           |           |           |  |
| anItem            | 275*( 2)  | 278 ( 2)  | 280 ( 2)  | 552*( 9)  | 556*( 9)  | 562*( 9)  | 569*( 9)  | 576*( 9)  |  |
|                   | 579*( 9)  | 582*( 9)  | 591*( 9)  | 597*( 9)  | 606*( 9)  | 611*( 9)  | 2485*(10) | 2507*(10) |  |
|                   | 2520*(10) | 2522 (10) | 2558*(10) | 2573*(10) | 2589*(10) | 2591 (10) | 2611*(10) | 2640*(10) |  |
|                   | 2642 (10) | 2677*(10) | 2683 (10) | 2721*(10) | 2726 (10) | 2737*(10) | 2740 (10) | 283*(11)  |  |
|                   | 726*(12)  | 736*(12)  | 738 (12)  | 151*(30)  | 156 (30)  | 159 (30)  |           |           |  |
| anObject          | 317*(12)  | 326=(12)  | 327 (12)  | 329 (12)  | 514*(12)  | 523=(12)  | 524 (12)  | 526 (12)  |  |
| anObjName         | 130*(31)  | 105*(32)  | 109=(32)  | 109 (32)  | 110 (32)  | 116 (32)  | 124 (32)  |           |  |
| anScrollBar       | 211*(20)  | 227 (20)  | 228 (20)  | 229 (20)  | 239 (20)  | 240 (20)  | 241 (20)  |           |  |
| aNumber           | 1520*(10) | 1560=(10) | 1570 (10) | 1584 (10) | 1620*(10) | 1660=(10) | 1670 (10) | 1684 (10) |  |
|                   | 323*(25)  | 738*(26)  | 741 (26)  |           |           |           |           |           |  |
| aNumberText       | 26*( 6)   | 43 ( 6)   | 43 ( 6)   | 43 ( 6)   |           |           |           |           |  |
| anyName           | 399*(26)  | 405=(26)  | 408=(26)  |           |           |           |           |           |  |
| AObjNameTbl       | 54 (31)   | 55*(31)   |           |           |           |           |           |           |  |
| aPageNumber       | 342*(36)  | 345*(36)  | 1125*(37) | 1167=(37) | 1170 (37) | 1172 (37) | 1569*(37) | 1593 (37) |  |
|                   | 1844*(37) | 1850 (37) | 1852 (37) | 1859 (37) | 1866 (37) |           |           |           |  |
| aPasteCommand     | 760*(40)  | 774 (40)  | 775 (40)  | 776 (40)  | 777 (40)  | 832*(40)  |           |           |  |
| aPatch            | 215*(35)  | 223=(35)  | 224 (35)  | 226=(35)  | 226 (35)  | 227 (35)  | 231 (35)  |           |  |
| aPattern          | 27*( 6)   | 44 ( 6)   | 44 ( 6)   | 44 ( 6)   |           |           |           |           |  |
| aPDownEvent       | 976*(13)  | 577*(16)  | 586 (16)  |           |           |           |           |           |  |
| ApFontID          | 359*(26)  | 374 (26)  |           |           |           |           |           |           |  |
| aPicture          | 22*( 6)   | 39 ( 6)   | 39 ( 6)   | 39 ( 6)   |           |           |           |           |  |

|                          |            |            |            |            |            |            |            |            |  |
|--------------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| aPoint                   | 388* ( 9)  | 1452* (10) | 1456 (10)  | 1457 (10)  | 1474 (10)  | 1479 (10)  | 1488 (10)  | 1496 (10)  |  |
|                          | 2072* (10) | 2078 (10)  | 2079 (10)  | 2081 (10)  | 2082 (10)  | 2089 (10)  | 2095 (10)  | 2095 (10)  |  |
|                          | 2096 (10)  | 2098 (10)  | 2098 (10)  | 2104 (10)  | 2112 (10)  | 2118 (10)  | 2118 (10)  | 2119 (10)  |  |
|                          | 2121 (10)  | 2121 (10)  | 2127 (10)  |            |            |            |            |            |  |
| aPopup                   | 23* ( 6)   | 40 ( 6)    | 40 ( 6)    | 40 ( 6)    |            |            |            |            |  |
| aPort                    | 1572* (37) | 1600 (37)  | 1601 (37)  | 985* (40)  | 991 (40)   | 992 (40)   |            |            |  |
| applEvt                  | 2614 (16)  |            |            |            |            |            |            |            |  |
| app2Evt                  | 2616 (16)  |            |            |            |            |            |            |            |  |
| app3Evt                  | 2618 (16)  |            |            |            |            |            |            |            |  |
| app4Evt                  | 654 (16)   | 1167 (16)  | 2620 (16)  |            |            |            |            |            |  |
| apParam                  | 581* (17)  | 587 (17)   |            |            |            |            |            |            |  |
| AppFile                  | 548 (13)   | 817 (13)   | 834 (13)   | 860 (13)   | 865 (13)   | 1213 (13)  | 329 (16)   | 408 (16)   |  |
|                          | 707 (16)   | 1435 (16)  | 2183 (16)  | 2430 (16)  | 799 (17)   | 958 (17)   |            |            |  |
| appleMenu                | 2847* (16) | 2858* (16) | 2859 (16)  |            |            |            |            |            |  |
| ApplFont                 | 747 (37)   |            |            |            |            |            |            |            |  |
| applFont                 | 263 (26)   | 372 (26)   | 362 (37)   | 365 (41)   |            |            |            |            |  |
| ApplicZone               | 361 (14)   | 2099 (16)  | 431 (28)   | 1092 (28)  |            |            |            |            |  |
| applZone                 | 336* (14)  | 361* (14)  | 362 (14)   | 363 (14)   | 365 (14)   | 1062* (28) | 1072 (28)  | 1092* (28) |  |
| appPrint                 | 640 (14)   |            |            |            |            |            |            |            |  |
| apRefnum                 | 580* (17)  | 587 (17)   |            |            |            |            |            |            |  |
| aPrintHandler            | 2433* (16) | 2456* (16) | 2457 (16)  | 2459 (16)  | 2461 (16)  |            |            |            |  |
| aPrintStyleChangeCommand | 1319* (37) | 1343 (37)  | 1344 (37)  | 1345 (37)  | 1352 (37)  | 1354 (37)  | 1358 (37)  |            |  |
| aPrJob                   | 1441* (37) | 1467* (37) | 1471 (37)  | 1472 (37)  | 1479 (37)  |            |            |            |  |
| aProc                    | 221* (27)  | 565* (28)  | 577 (28)   | 578 (28)   | 579 (28)   | 580 (28)   |            |            |  |
| aPtr                     | 241* (16)  | 244* (16)  | 245 (16)   | 19* (23)   | 28* (23)   | 29 (23)    | 45* (23)   | 50* (23)   |  |
|                          | 51 (23)    |            |            |            |            |            |            |            |  |
| aQDRect                  | 252* ( 9)  | 255* ( 9)  | 479* ( 9)  | 492* (10)  | 500 (10)   | 501 (10)   | 512* (10)  | 513 (10)   |  |
|                          | 513 (10)   | 514 (10)   | 514 (10)   | 516 (10)   | 516 (10)   | 517 (10)   | 518 (10)   | 519 (10)   |  |
|                          | 519 (10)   | 520 (10)   | 520 (10)   | 522 (10)   | 523 (10)   | 525 (10)   | 526 (10)   | 545* (10)  |  |
|                          | 567* (10)  | 691* (10)  | 723* (10)  | 731 (10)   | 731 (10)   | 733 (10)   | 733 (10)   | 735 (10)   |  |
|                          | 736 (10)   | 736 (10)   | 742* (10)  | 750 (10)   | 750 (10)   | 752 (10)   | 752 (10)   | 754 (10)   |  |
|                          | 755 (10)   | 755 (10)   | 767* (10)  | 777 (10)   | 777 (10)   | 778 (10)   | 778 (10)   | 779 (10)   |  |
|                          | 790 (10)   | 790 (10)   | 792 (10)   | 792 (10)   | 794 (10)   | 799 (10)   | 799 (10)   | 801 (10)   |  |
|                          | 801 (10)   | 807 (10)   | 809 (10)   | 809 (10)   | 811 (10)   | 811 (10)   | 823* (10)  | 2315* (10) |  |
|                          | 2323 (10)  | 2323 (10)  |            |            |            |            |            |            |  |
| aRadio                   | 19* ( 6)   | 36 ( 6)    | 36 ( 6)    | 36 ( 6)    |            |            |            |            |  |
| area                     | 271* ( 5)  | 323* ( 5)  | 369* ( 5)  | 414* ( 5)  | 472* ( 5)  | 474* ( 5)  | 476* ( 5)  | 532* ( 5)  |  |
|                          | 595* ( 5)  | 1245* ( 6) | 1294 ( 6)  | 1531* ( 6) | 1556 ( 6)  | 1750* ( 6) | 1776 ( 6)  | 1950* ( 6) |  |
|                          | 1972 ( 6)  | 2291* ( 6) | 2306 ( 6)  | 2315 ( 6)  | 2320 ( 6)  | 2328 ( 6)  | 2335* ( 6) | 2347 ( 6)  |  |
|                          | 2347 ( 6)  | 2358* ( 6) | 2379 ( 6)  | 2803* ( 6) | 2826 ( 6)  | 2879* ( 6) | 2890 ( 6)  | 2891 ( 6)  |  |
|                          | 2892 ( 6)  | 3113* ( 6) | 3121 ( 6)  | 3126 ( 6)  | 3285* ( 6) | 3297 ( 6)  | 3298 ( 6)  | 3299 ( 6)  |  |
|                          | 227* ( 9)  | 235* ( 9)  | 236* ( 9)  | 329* (10)  | 339* (10)  | 686* (10)  | 702 (10)   | 717 (10)   |  |
|                          | 719 (10)   | 723 (10)   | 742 (10)   | 1592* (13) | 1596* (13) | 1695* (13) | 1981* (13) | 2102* (13) |  |
|                          | 2178* (13) | 2307* (13) | 2401* (13) | 2572* (13) | 394* (18)  | 406 (18)   | 414 (18)   | 485 (18)   |  |
|                          | 495 (18)   | 497 (18)   | 504 (18)   | 756* (18)  | 759 (18)   | 863* (18)  | 444* (19)  | 276* (21)  |  |
|                          | 278 (21)   | 281 (21)   | 317* (21)  | 690* (21)  | 965* (21)  | 970 (21)   | 971 (21)   | 975 (21)   |  |
|                          | 1384* (21) | 1397 (21)  | 229* (23)  | 90* (24)   | 424* (36)  | 712* (37)  | 753 (37)   | 218* (38)  |  |
|                          | 982* (40)  | 1009 (40)  | 1029 (40)  |            |            |            |            |            |  |
| aRect                    | 2294* ( 6) | 2305 ( 6)  | 2306 ( 6)  | 2306 ( 6)  | 2312 ( 6)  | 2313 ( 6)  | 2318 ( 6)  | 265* ( 9)  |  |
|                          | 269* ( 9)  | 270* ( 9)  | 395* (10)  | 410 (10)   | 414 (10)   | 416 (10)   | 432* (10)  | 449 (10)   |  |
|                          | 471 (10)   | 490* (10)  | 499 (10)   | 500 (10)   | 522 (10)   | 524 (10)   | 525 (10)   | 546* (10)  |  |
|                          | 565* (10)  | 689* (10)  | 714 (10)   | 716 (10)   | 717 (10)   | 845* (10)  | 876 (10)   | 925 (10)   |  |
|                          | 942* (10)  | 973 (10)   | 1022 (10)  | 1406* (10) | 1409 (10)  | 1410 (10)  | 1423* (10) | 1432 (10)  |  |
|                          | 1434 (10)  | 1435 (10)  | 1451* (10) | 1458 (10)  | 1459 (10)  | 1474 (10)  | 1479 (10)  | 1488 (10)  |  |
|                          | 1496 (10)  | 1521* (10) | 1603 (10)  | 1604 (10)  | 1621* (10) | 1703 (10)  | 1704 (10)  | 1786* (10) |  |
|                          | 1803 (10)  | 1825 (10)  | 2935* (10) | 2949 (10)  | 2950 (10)  | 1004* (18) | 1025* (18) | 1026 (18)  |  |
|                          | 250* (21)  | 252 (21)   | 253 (21)   | 354* (21)  | 356 (21)   | 357 (21)   | 781* (21)  | 783 (21)   |  |
|                          | 784 (21)   | 784 (21)   | 785 (21)   | 785 (21)   | 785 (21)   | 785 (21)   | 78* (23)   | 90 (23)    |  |
|                          | 91 (23)    | 91 (23)    | 102 (23)   | 102 (23)   | 232* (23)  | 243 (23)   | 245 (23)   | 253 (23)   |  |
|                          | 256* (23)  | 259 (23)   | 259 (23)   | 259 (23)   | 260 (23)   | 263 (23)   | 268 (23)   | 932* (37)  |  |
|                          | 939* (37)  | 948 (37)   | 948 (37)   | 949 (37)   | 949 (37)   | 961 (37)   | 677* (40)  | 1576* (40) |  |
| aRefNum                  | 1179* (13) | 1181* (13) | 2027* (16) | 2039 (16)  | 2040 (16)  | 2078* (16) | 2104* (16) | 2115* (16) |  |
|                          | 365* (17)  | 379 (17)   | 410* (17)  | 424 (17)   |            |            |            |            |  |
| areMinimalMargins        | 366* (36)  | 1046* (37) | 1048 (37)  |            |            |            |            |            |  |
| aResType                 | 2895* (13) | 1117* (14) | 1122 (14)  |            |            |            |            |            |  |
| aRgn                     | 302* ( 9)  | 1138* (10) | 1145 (10)  | 1152 (10)  | 816* (14)  | 818* (14)  | 819 (14)   | 820 (14)   |  |
|                          | 1555* (18) | 1579* (18) | 1582 (18)  | 1592 (18)  | 1597 (18)  | 506* (19)  |            |            |  |
| aRow                     | 236* ( 9)  | 270* ( 9)  | 280* ( 9)  | 309* ( 9)  | 316* ( 9)  | 320* ( 9)  | 326* ( 9)  | 363* ( 9)  |  |
|                          | 715* ( 9)  | 761* ( 9)  | 339* (10)  | 624* (10)  | 633 (10)   | 635 (10)   | 651 (10)   | 669 (10)   |  |
|                          | 936* (10)  | 950 (10)   | 950 (10)   | 956 (10)   | 965 (10)   | 967 (10)   | 973 (10)   | 973 (10)   |  |
|                          | 974 (10)   | 1247* (10) | 1255 (10)  | 1255 (10)  | 1261* (10) | 1263 (10)  | 1272 (10)  | 1274 (10)  |  |
|                          | 1274 (10)  | 1280 (10)  | 1295 (10)  | 1297 (10)  | 1312 (10)  | 1314 (10)  | 1318 (10)  | 1373* (10) |  |
|                          | 1381 (10)  | 1381 (10)  | 1384 (10)  | 1392 (10)  | 1449* (10) | 1460* (10) | 1502* (10) | 1502 (10)  |  |
|                          | 1615* (10) | 1627 (10)  | 1633 (10)  | 1642 (10)  | 1650 (10)  | 1662 (10)  | 1664 (10)  | 1703 (10)  |  |
|                          | 1703 (10)  | 1786* (10) | 1795 (10)  | 1795 (10)  | 1799 (10)  | 1812 (10)  | 1817 (10)  | 1820 (10)  |  |
|                          | 1820 (10)  | 1908* (10) | 1911 (10)  | 1911 (10)  | 1914 (10)  | 1922 (10)  | 1923 (10)  | 3080* (10) |  |
|                          | 3083 (10)  | 3147* (10) | 3150 (10)  |            |            |            |            |            |  |
| aRowSelectCommand        | 626* (10)  | 649 (10)   | 650 (10)   | 651 (10)   | 653 (10)   |            |            |            |  |
| arrow                    | 124 ( 4)   | 257 ( 4)   | 803 (14)   | 1594 (16)  | 3090 (16)  | 282 (19)   |            |            |  |
| ArrTypeList              | 552 (13)   | 553* (13)  |            |            |            |            |            |            |  |
| asBoolean                | 867* (26)  | 940 (26)   |            |            |            |            |            |            |  |
| asByte                   | 889* (26)  | 999 (26)   |            |            |            |            |            |            |  |
| ascent                   | 1265 ( 6)  | 2360 (10)  | 2361 (10)  | 326 (40)   | 451 (40)   | 1416 (40)  | 1417 (40)  |            |  |
| asChar                   | 878* (26)  | 966 (26)   |            |            |            |            |            |            |  |
| asCntlAdornment          | 898* (26)  | 927 (26)   | 1069 (26)  |            |            |            |            |            |  |
| aScrollBar               | 474* (14)  | 694 (14)   | 694 (14)   | 694 (14)   | 152* (20)  | 160* (20)  | 161 (20)   | 172 (20)   |  |
|                          | 173 (20)   | 14* (21)   | 20* (21)   | 22 (21)    | 25 (21)    |            |            |            |  |
| aScroller                | 475* (14)  | 695 (14)   | 695 (14)   | 695 (14)   | 834* (14)  | 865 (14)   | 866 (14)   | 867 (14)   |  |
|                          | 869 (14)   | 901* (14)  | 918* (14)  | 938 (14)   | 939 (14)   | 940 (14)   | 942 (14)   | 944 (14)   |  |
|                          | 1193* (18) | 1199* (18) | 1200 (18)  | 1201 (18)  |            |            |            |            |  |
| asHandle                 | 885* (26)  | 976 (26)   | 979 (26)   |            |            |            |            |            |  |
| asInteger                | 873* (26)  | 942 (26)   | 944 (26)   | 948 (26)   | 952 (26)   | 954 (26)   | 1001 (26)  |            |  |
| aSize                    | 1080* (40) | 1107* (40) | 1113* (40) | 1117 (40)  | 1122 (40)  | 1143* (40) | 1151 (40)  | 1170 (40)  |  |
| askForFilename           | 1293* (13) | 1037* (17) | 1071 (17)  | 1086 (17)  | 1098 (17)  |            |            |            |  |
| asLongInt                | 876* (26)  | 946 (26)   | 950 (26)   | 957 (26)   | 958 (26)   |            |            |            |  |
| asObject                 | 888* (26)  | 997 (26)   |            |            |            |            |            |            |  |
| asOSType                 | 890* (26)  | 1006 (26)  |            |            |            |            |            |            |  |
| asPattern                | 891* (26)  | 1013 (26)  |            |            |            |            |            |            |  |
| asPoint                  | 886* (26)  | 982 (26)   | 983 (26)   |            |            |            |            |            |  |

|                      |           |           |           |           |           |           |           |           |  |
|----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| asPointer            | 881*(26)  | 971 (26)  |           |           |           |           |           |           |  |
| asRect               | 887*(26)  | 988 (26)  | 989 (26)  | 991 (26)  | 993 (26)  |           |           |           |  |
| asRGBColor           | 892*(26)  | 1018 (26) | 1025 (26) | 1026 (26) | 1028 (26) |           |           |           |  |
| asStrHandle          | 897*(26)  | 1062 (26) | 1065 (26) |           |           |           |           |           |  |
| asString             | 877*(26)  | 962 (26)  |           |           |           |           |           |           |  |
| asStyle              | 893*(26)  | 913 (26)  |           |           |           |           |           |           |  |
| aStaticText          | 24*( 6)   | 41 ( 6)   | 41 ( 6)   | 41 ( 6)   |           |           |           |           |  |
| aStdPrintHandler     | 176*(37)  | 199 (37)  | 200 (37)  | 201 (37)  | 202 (37)  | 203 (37)  |           |           |  |
| astr                 | 223*(25)  |           |           |           |           |           |           |           |  |
| aString              | 1123*( 6) | 1142 ( 6) | 1145 ( 6) | 2672*( 6) | 2692 ( 6) | 2695 ( 6) | 3318*( 6) | 3360*( 6) |  |
|                      | 3366 ( 6) | 3369 ( 6) | 3398*( 6) | 3404 ( 6) | 3405 ( 6) | 3418*( 6) | 3425 ( 6) | 3426 ( 6) |  |
|                      | 3479*( 6) | 3483 ( 6) | 3484 ( 6) | 3496*( 6) | 3500 ( 6) | 3501 ( 6) | 469*( 9)  | 552*( 9)  |  |
|                      | 588*( 9)  | 2389*(10) | 2611*(10) | 2627*(10) | 2629 (10) | 625*(12)  | 637 (12)  | 638-(12)  |  |
|                      | 638 (12)  | 639 (12)  | 21*(19)   | 65 (19)   | 66 (19)   | 67 (19)   | 90*(19)   | 141 (19)  |  |
|                      | 142 (19)  | 143 (19)  | 722*(19)  | 730-(19)  | 731 (19)  | 732 (19)  | 903*(26)  | 957 (26)  |  |
|                      | 959 (26)  | 982 (26)  | 984 (26)  | 988 (26)  | 990 (26)  | 991 (26)  | 992 (26)  | 993 (26)  |  |
|                      | 994 (26)  | 1026 (26) | 1027 (26) | 1028 (26) | 1029 (26) | 1047 (26) | 1049 (26) | 1053 (26) |  |
|                      | 1055 (26) | 1056 (26) | 1057 (26) | 1058 (26) | 1059 (26) | 318*(30)  | 333 (30)  | 334 (30)  |  |
|                      | 779*(37)  | 805 (37)  | 807 (37)  | 809 (37)  | 811 (37)  | 813 (37)  | 814 (37)  | 815 (37)  |  |
|                      | 817 (37)  | 818 (37)  | 819 (37)  |           |           |           |           |           |  |
| aStrip               | 1252*(37) | 1266 (37) |           |           |           |           |           |           |  |
| aStyle               | 355*(25)  | 1094*(26) | 1100 (26) | 1101 (26) | 1102 (26) | 1103 (26) | 265*(29)  | 596*(30)  |  |
|                      | 603 (30)  | 608 (30)  | 36*(38)   | 185*(38)  | 277*(38)  | 544*(40)  | 555 (40)  | 795*(40)  |  |
|                      | 803 (40)  |           |           |           |           |           |           |           |  |
| aSubview             | 701*( 6)  | 704-( 6)  | 705 ( 6)  | 705 ( 6)  | 706 ( 6)  |           |           |           |  |
| asVCoordinate        | 894*(26)  | 1044 (26) |           |           |           |           |           |           |  |
| asVPoint             | 895*(26)  | 1047 (26) | 1048 (26) |           |           |           |           |           |  |
| asVRect              | 896*(26)  | 1053 (26) | 1054 (26) | 1056 (26) | 1058 (26) |           |           |           |  |
| At                   | 81*(11)   | 76*(12)   | 94-(12)   | 305 (12)  | 502 (12)  | 697 (12)  | 755 (12)  |           |  |
| atDrvvrVersNum       | 179*(25)  |           |           |           |           |           |           |           |  |
| aTEH                 | 613*(38)  | 52*(39)   | 61 (39)   | 64 (39)   | 65 (39)   | 67 (39)   | 68 (39)   | 73 (39)   |  |
|                      | 83 (39)   | 84 (39)   | 86 (39)   |           |           |           |           |           |  |
| aTEHandle            | 79*(23)   | 91-(23)   | 92 (23)   | 94 (23)   | 99 (23)   | 100 (23)  | 107 (23)  |           |  |
| aTEStyleCommand      | 798*(40)  | 801 (40)  | 802 (40)  | 803 (40)  | 804 (40)  |           |           |           |  |
| aTEView              | 23*(39)   | 27 (39)   | 27 (39)   | 27 (39)   |           |           |           |           |  |
| aTextStyle           | 130*(21)  | 148 (21)  | 149 (21)  | 73*(40)   | 98 (40)   | 100 (40)  | 710*(41)  | 766-(41)  |  |
|                      | 767-(41)  |           |           |           |           |           |           |           |  |
| aTitle               | 1398*(13) | 1380*(17) | 1387 (17) | 1390 (17) | 355*(25)  | 1094*(26) | 1099 (26) | 1993*(37) |  |
|                      | 2000 (37) | 2001 (37) |           |           |           |           |           |           |  |
| AtPut                | 179*(11)  | 102*(12)  |           |           |           |           |           |           |  |
| aTrackPhase          | 690*( 9)  | 3028*(10) | 3034 (10) | 2131*(13) | 2492*(13) | 2534*(13) | 3216*(16) | 3231 (16) |  |
|                      | 75*(21)   | 80 (21)   | 82 (21)   | 469*(21)  | 473 (21)  | 97*(22)   |           |           |  |
| AttachPrintHandler   | 1671*(13) | 523*(18)  | 292 (37)  | 325 (37)  |           |           |           |           |  |
| aType                | 1663*(13) | 669*(18)  | 674 (18)  | 256*(38)  | 563*(40)  | 565 (40)  |           |           |  |
| aTypingCommand       | 697*(40)  | 742-(40)  | 743 (40)  | 744 (40)  | 814*(40)  | 817 (40)  | 818 (40)  | 819 (40)  |  |
|                      | 820 (40)  |           |           |           |           |           |           |           |  |
| autoKey              | 648 (16)  | 1404 (16) | 2575 (16) | 2579 (16) |           |           |           |           |  |
| automatic            | 1676*(13) | 1699*(13) | 2356*(13) | 2405*(13) | 708*(18)  | 710 (18)  | 767*(18)  | 769 (18)  |  |
|                      | 25*(24)   | 101*(24)  | 219*(36)  | 232*(36)  | 428*(36)  | 408*(37)  | 416-(37)  | 445*(37)  |  |
|                      | 725*(37)  | 735 (37)  | 770*(37)  | 840*(37)  | 845*(37)  | 855-(37)  | 860 (37)  | 861 (37)  |  |
|                      | 865 (37)  | 973*(37)  | 997 (37)  | 1251*(37) | 1265 (37) | 336*(38)  | 574*(40)  | 586-(40)  |  |
| AutoScroll           | 1843*(13) | 2519*(13) | 3303 (16) | 3318 (16) | 182*(20)  | 34*(22)   | 198 (39)  |           |  |
| AutoScrolling        | 624 ( 6)  | 267*(38)  | 201*(40)  |           |           |           |           |           |  |
| autoScrollLimit      | 3133 (16) |           |           |           |           |           |           |           |  |
| autoscrollLimit      | 3117*(16) | 3135 (16) | 3136 (16) | 3300 (16) |           |           |           |           |  |
| AutoscrollTEView     | 608*(38)  | 176*(39)  | 228-(39)  | 54 (40)   | 110 (40)  |           |           |           |  |
| avail                | 226*(28)  | 257-(28)  | 261-(28)  | 264 (28)  | 267 (28)  | 273 (28)  |           |           |  |
| aVertexSelectCommand | 628*(10)  | 667 (10)  | 668 (10)  | 669 (10)  | 671 (10)  |           |           |           |  |
| aView                | 1227*( 6) | 1229 ( 6) | 1230 ( 6) | 1231 ( 6) | 1331*( 6) | 1333 ( 6) | 1333 ( 6) | 681*( 9)  |  |
|                      | 2869*(10) | 2882 (10) | 1409*(13) | 1421*(13) | 60*(15)   | 70-(15)   | 75-(15)   | 77-(15)   |  |
|                      | 80 (15)   | 82 (15)   | 109*(15)  | 176-(15)  | 180-(15)  | 182 (15)  | 185 (15)  | 187 (15)  |  |
|                      | 2751*(16) | 2754 (16) | 125*(17)  | 127 (17)  | 436*(17)  | 1432*(17) | 1434 (17) | 1414*(18) |  |
|                      | 1417-(18) | 1418 (18) | 1420 (18) | 1421-(18) | 1421 (18) | 1816*(18) | 1819-(18) | 1820 (18) |  |
|                      | 1822 (18) | 1823-(18) | 1823 (18) | 509*(37)  | 512 (37)  |           |           |           |  |
| aViewForClipboard    | 2489*(16) | 2498-(16) | 2507-(16) | 2508 (16) | 2511 (16) |           |           |           |  |
| AWatchTask           | 48*( 4)   | 137 ( 4)  |           |           |           |           |           |           |  |
| aWidth               | 325*( 9)  | 331*( 9)  | 337*( 9)  | 362*( 9)  | 616*( 9)  | 1515*(10) | 1543 (10) | 1551 (10) |  |
|                      | 1576 (10) | 1593 (10) | 1600 (10) | 1715*(10) | 1717 (10) | 1739*(10) | 1741 (10) | 1882*(10) |  |
|                      | 1894 (10) | 1897 (10) | 2749*(10) | 2751 (10) |           |           |           |           |  |
| aWindow              | 897*(13)  | 904*(13)  | 910*(13)  | 1412*(13) | 1426*(13) | 476*(14)  | 693 (14)  | 693 (14)  |  |
|                      | 693 (14)  | 833*(14)  | 845 (14)  | 849-(14)  | 851 (14)  | 856 (14)  | 862 (14)  | 863 (14)  |  |
|                      | 867 (14)  | 872 (14)  | 874 (14)  | 877 (14)  | 884 (14)  | 886 (14)  | 900*(14)  | 912 (14)  |  |
|                      | 916-(14)  | 925 (14)  | 940 (14)  | 948 (14)  | 951 (14)  | 954 (14)  | 961 (14)  | 963 (14)  |  |
|                      | 977*(14)  | 1001-(14) | 1003 (14) | 1004 (14) | 1007 (14) | 1009 (14) | 261*(16)  | 263 (16)  |  |
|                      | 542*(16)  | 548-(16)  | 549 (16)  | 550 (16)  | 706*(16)  | 872*(16)  | 1060*(16) | 1306*(16) |  |
|                      | 1316-(16) | 1317 (16) | 1318 (16) | 1551*(16) | 1573-(16) | 1576 (16) | 1608 (16) | 1612 (16) |  |
|                      | 1614 (16) | 1621 (16) | 1624 (16) | 1632 (16) | 1663*(16) | 1688-(16) | 1689 (16) | 1690 (16) |  |
|                      | 1706*(16) | 1717-(16) | 1718 (16) | 1719 (16) | 2135*(16) | 2161-(16) | 2163 (16) | 2165 (16) |  |
|                      | 2166 (16) | 2324*(16) | 146*(17)  | 148 (17)  | 188*(17)  | 190 (17)  | 454*(17)  | 1385*(17) |  |
|                      | 1387 (17) | 1449*(17) | 1451 (17) | 1452 (17) | 404*(19)  | 1011*(37) | 1019-(37) | 1020 (37) |  |
|                      | 1021 (37) |           |           |           |           |           |           |           |  |
| aWmgrWindow          | 758*(13)  | 916*(13)  | 927*(13)  | 548 (16)  | 1552*(16) | 1573 (16) | 1662*(16) | 1680-(16) |  |
|                      | 1682 (16) | 1686 (16) | 1688 (16) | 1871*(16) | 1873 (16) | 1874 (16) | 2734*(16) | 2737 (16) |  |
|                      | 3003*(16) | 3027 (16) | 3031 (16) | 3375*(16) | 3378 (16) | 3378 (16) | 3380 (16) | 3383 (16) |  |
|                      | 87*(19)   | 120-(19)  | 123-(19)  | 125 (19)  | 141 (19)  | 143 (19)  |           |           |  |
| aWmgrWindow          | 962*(13)  | 976*(14)  | 991 (14)  | 996-(14)  | 1007 (14) | 540*(16)  | 544 (16)  | 545 (16)  |  |
|                      | 552 (16)  | 1227*(16) | 1276-(16) | 1278-(16) | 1280 (16) | 1286 (16) | 1291 (16) | 1296 (16) |  |
|                      | 1565 (16) | 1569 (16) | 1583 (16) | 1605 (16) | 1627 (16) | 1628 (16) | 1631 (16) |           |  |
| aZone                | 412*(28)  | 431-(28)  | 432 (28)  | 432 (28)  |           |           |           |           |  |
| -B-                  |           |           |           |           |           |           |           |           |  |
| b                    | 297*(25)  | 300*(25)  | 461*(25)  | 479*(25)  | 500*(26)  | 502 (26)  | 503 (26)  | 514*(26)  |  |
|                      | 516 (26)  | 519 (26)  | 1244*(26) | 1247 (26) | 1257*(26) | 1261 (26) | 869*(28)  | 62*(37)   |  |
|                      | 77-(37)   | 78 (37)   | 231*(37)  | 235 (37)  | 1279*(37) | 1296 (37) |           |           |  |
| b0                   | 1688*(37) |           |           |           |           |           |           |           |  |
| b1                   | 1688*(37) | 1726 (37) |           |           |           |           |           |           |  |
| BackColor            | 137 ( 6)  |           |           |           |           |           |           |           |  |
| backP                | 45*(32)   | 62-(32)   | 68-(32)   | 68 (32)   | 69 (32)   | 73-(32)   | 73 (32)   | 75 (32)   |  |
|                      | 78-(32)   | 78 (32)   | 81-(32)   | 81 (32)   | 82 (32)   |           |           |           |  |

[illegible]



|      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 303  | (2)  | 318  | (2)  | 74   | (3)  | 79   | (3)  | 10   | (4)  | 17   | (4)  | 20   | (4)  | 26   | (4)  |
| 27   | (4)  | 70   | (4)  | 88   | (4)  | 62   | (5)  | 84   | (5)  | 98   | (5)  | 118  | (5)  | 164  | (5)  |
| 173  | (5)  | 188  | (5)  | 190  | (5)  | 192  | (5)  | 194  | (5)  | 204  | (5)  | 213  | (5)  | 228  | (5)  |
| 230  | (5)  | 232  | (5)  | 234  | (5)  | 281  | (5)  | 295  | (5)  | 301  | (5)  | 308  | (5)  | 327  | (5)  |
| 341  | (5)  | 347  | (5)  | 354  | (5)  | 373  | (5)  | 418  | (5)  | 484  | (5)  | 487  | (5)  | 530  | (5)  |
| 541  | (5)  | 543  | (5)  | 593  | (5)  | 602  | (5)  | 604  | (5)  | 606  | (5)  | 610  | (5)  | 612  | (5)  |
| 614  | (5)  | 616  | (5)  | 618  | (5)  | 620  | (5)  | 660  | (5)  | 662  | (5)  | 679  | (5)  | 681  | (5)  |
| 148  | (6)  | 392  | (6)  | 509  | (6)  | 698  | (6)  | 716  | (6)  | 863  | (6)  | 947  | (6)  | 956  | (6)  |
| 965  | (6)  | 974  | (6)  | 991  | (6)  | 1076 | (6)  | 1085 | (6)  | 1094 | (6)  | 1103 | (6)  | 1331 | (6)  |
| 1348 | (6)  | 1392 | (6)  | 1577 | (6)  | 1615 | (6)  | 1797 | (6)  | 1989 | (6)  | 2460 | (6)  | 2484 | (6)  |
| 2576 | (6)  | 2616 | (6)  | 2779 | (6)  | 2805 | (6)  | 2866 | (6)  | 2877 | (6)  | 3102 | (6)  | 3218 | (6)  |
| 3231 | (6)  | 3256 | (6)  | 3267 | (6)  | 3281 | (6)  | 3312 | (6)  | 3330 | (6)  | 3357 | (6)  | 3379 | (6)  |
| 3493 | (6)  | 3509 | (6)  | 3516 | (6)  | 168  | (7)  | 170  | (7)  | 233  | (7)  | 268  | (8)  | 99   | (9)  |
| 100  | (9)  | 101  | (9)  | 142  | (9)  | 143  | (9)  | 147  | (9)  | 148  | (9)  | 149  | (9)  | 150  | (9)  |
| 179  | (9)  | 180  | (9)  | 183  | (9)  | 240  | (9)  | 244  | (9)  | 275  | (9)  | 307  | (9)  | 310  | (9)  |
| 350  | (9)  | 356  | (9)  | 359  | (9)  | 366  | (9)  | 370  | (9)  | 374  | (9)  | 381  | (9)  | 385  | (9)  |
| 434  | (9)  | 435  | (9)  | 438  | (9)  | 482  | (9)  | 527  | (9)  | 528  | (9)  | 531  | (9)  | 556  | (9)  |
| 565  | (9)  | 597  | (9)  | 600  | (9)  | 603  | (9)  | 606  | (9)  | 641  | (9)  | 642  | (9)  | 647  | (9)  |
| 654  | (9)  | 674  | (9)  | 675  | (9)  | 678  | (9)  | 682  | (9)  | 687  | (9)  | 692  | (9)  | 715  | (9)  |
| 738  | (9)  | 761  | (9)  | 37   | (10) | 38   | (10) | 41   | (10) | 367  | (10) | 381  | (10) | 488  | (10) |
| 849  | (10) | 946  | (10) | 1164 | (10) | 1248 | (10) | 1762 | (10) | 1841 | (10) | 1852 | (10) | 1933 | (10) |
| 1952 | (10) | 1985 | (10) | 2045 | (10) | 2058 | (10) | 2206 | (10) | 2207 | (10) | 2210 | (10) | 2334 | (10) |
| 2446 | (10) | 2447 | (10) | 2450 | (10) | 2496 | (10) | 2507 | (10) | 2677 | (10) | 2693 | (10) | 2710 | (10) |
| 2721 | (10) | 2779 | (10) | 2821 | (10) | 2854 | (10) | 2870 | (10) | 2931 | (10) | 3030 | (10) | 3080 | (10) |
| 3113 | (10) | 3147 | (10) | 132  | (11) | 154  | (11) | 24   | (12) | 313  | (12) | 425  | (12) | 510  | (12) |
| 736  | (12) | 503  | (13) | 505  | (13) | 506  | (13) | 507  | (13) | 508  | (13) | 509  | (13) | 510  | (13) |
| 616  | (13) | 622  | (13) | 656  | (13) | 658  | (13) | 704  | (13) | 758  | (13) | 777  | (13) | 780  | (13) |
| 795  | (13) | 805  | (13) | 817  | (13) | 834  | (13) | 865  | (13) | 922  | (13) | 937  | (13) | 951  | (13) |
| 1006 | (13) | 1049 | (13) | 1090 | (13) | 1096 | (13) | 1108 | (13) | 1110 | (13) | 1112 | (13) | 1117 | (13) |
| 1119 | (13) | 1121 | (13) | 1123 | (13) | 1141 | (13) | 1142 | (13) | 1174 | (13) | 1180 | (13) | 1181 | (13) |
| 1190 | (13) | 1193 | (13) | 1214 | (13) | 1232 | (13) | 1249 | (13) | 1249 | (13) | 1265 | (13) | 1266 | (13) |
| 1285 | (13) | 1293 | (13) | 1302 | (13) | 1317 | (13) | 1333 | (13) | 1380 | (13) | 1391 | (13) | 1443 | (13) |
| 1444 | (13) | 1469 | (13) | 1470 | (13) | 1523 | (13) | 1526 | (13) | 1542 | (13) | 1566 | (13) | 1568 | (13) |
| 1574 | (13) | 1578 | (13) | 1582 | (13) | 1584 | (13) | 1611 | (13) | 1613 | (13) | 1615 | (13) | 1619 | (13) |
| 1623 | (13) | 1633 | (13) | 1641 | (13) | 1653 | (13) | 1655 | (13) | 1657 | (13) | 1659 | (13) | 1663 | (13) |
| 1676 | (13) | 1699 | (13) | 1745 | (13) | 1746 | (13) | 1751 | (13) | 1752 | (13) | 1763 | (13) | 1772 | (13) |
| 1795 | (13) | 1809 | (13) | 1813 | (13) | 1819 | (13) | 1822 | (13) | 1826 | (13) | 1833 | (13) | 1839 | (13) |
| 1841 | (13) | 1846 | (13) | 1850 | (13) | 1868 | (13) | 1869 | (13) | 1870 | (13) | 1871 | (13) | 1872 | (13) |
| 1873 | (13) | 1874 | (13) | 1875 | (13) | 1876 | (13) | 1877 | (13) | 1878 | (13) | 1879 | (13) | 1880 | (13) |
| 1887 | (13) | 1888 | (13) | 1889 | (13) | 1890 | (13) | 1891 | (13) | 1892 | (13) | 1893 | (13) | 1894 | (13) |
| 1895 | (13) | 1896 | (13) | 1897 | (13) | 1898 | (13) | 1899 | (13) | 1900 | (13) | 1913 | (13) | 1914 | (13) |
| 1915 | (13) | 1916 | (13) | 1917 | (13) | 1918 | (13) | 1919 | (13) | 1920 | (13) | 1921 | (13) | 1922 | (13) |
| 1923 | (13) | 1924 | (13) | 1925 | (13) | 1926 | (13) | 1932 | (13) | 1957 | (13) | 1961 | (13) | 1967 | (13) |
| 1973 | (13) | 1975 | (13) | 1990 | (13) | 1994 | (13) | 2000 | (13) | 2002 | (13) | 2012 | (13) | 2048 | (13) |
| 2049 | (13) | 2050 | (13) | 2051 | (13) | 2063 | (13) | 2064 | (13) | 2065 | (13) | 2066 | (13) | 2087 | (13) |
| 2091 | (13) | 2097 | (13) | 2104 | (13) | 2108 | (13) | 2110 | (13) | 2112 | (13) | 2114 | (13) | 2116 | (13) |
| 2118 | (13) | 2120 | (13) | 2129 | (13) | 2133 | (13) | 2135 | (13) | 2173 | (13) | 2188 | (13) | 2190 | (13) |
| 2192 | (13) | 2194 | (13) | 2196 | (13) | 2198 | (13) | 2200 | (13) | 2202 | (13) | 2257 | (13) | 2259 | (13) |
| 2261 | (13) | 2263 | (13) | 2302 | (13) | 2356 | (13) | 2393 | (13) | 2395 | (13) | 2405 | (13) | 2439 | (13) |
| 2442 | (13) | 2443 | (13) | 2445 | (13) | 2451 | (13) | 2454 | (13) | 2456 | (13) | 2487 | (13) | 2494 | (13) |
| 2536 | (13) | 2539 | (13) | 2552 | (13) | 2553 | (13) | 2573 | (13) | 2592 | (13) | 2596 | (13) | 2598 | (13) |
| 2619 | (13) | 2633 | (13) | 2635 | (13) | 2641 | (13) | 2645 | (13) | 2663 | (13) | 2671 | (13) | 2678 | (13) |
| 2684 | (13) | 2685 | (13) | 2688 | (13) | 2736 | (13) | 2749 | (13) | 2754 | (13) | 2755 | (13) | 2785 | (13) |
| 2798 | (13) | 2800 | (13) | 2805 | (13) | 2830 | (13) | 2847 | (13) | 2852 | (13) | 2860 | (13) | 2876 | (13) |
| 2884 | (13) | 2911 | (13) | 2979 | (13) | 89   | (14) | 144  | (14) | 164  | (14) | 171  | (14) | 176  | (14) |
| 273  | (14) | 715  | (14) | 721  | (14) | 830  | (14) | 898  | (14) | 978  | (14) | 979  | (14) | 1063 | (14) |
| 1135 | (14) | 1243 | (14) | 214  | (15) | 262  | (15) | 287  | (15) | 313  | (15) | 190  | (16) | 319  | (16) |
| 329  | (16) | 408  | (16) | 703  | (16) | 1022 | (16) | 1120 | (16) | 1123 | (16) | 1211 | (16) | 1218 | (16) |
| 1219 | (16) | 1231 | (16) | 1337 | (16) | 1437 | (16) | 1550 | (16) | 1554 | (16) | 1660 | (16) | 1661 | (16) |
| 1665 | (16) | 1749 | (16) | 1813 | (16) | 1823 | (16) | 1834 | (16) | 1871 | (16) | 1952 | (16) | 2032 | (16) |
| 2033 | (16) | 2053 | (16) | 2247 | (16) | 2393 | (16) | 2394 | (16) | 2430 | (16) | 2434 | (16) | 2518 | (16) |
| 2801 | (16) | 2848 | (16) | 2998 | (16) | 3002 | (16) | 3009 | (16) | 3104 | (16) | 3110 | (16) | 3112 | (16) |
| 3114 | (16) | 3189 | (16) | 3216 | (16) | 11   | (17) | 12   | (17) | 137  | (17) | 156  | (17) | 242  | (17) |
| 275  | (17) | 366  | (17) | 410  | (17) | 516  | (17) | 517  | (17) | 605  | (17) | 615  | (17) | 616  | (17) |
| 721  | (17) | 784  | (17) | 787  | (17) | 799  | (17) | 803  | (17) | 804  | (17) | 805  | (17) | 902  | (17) |
| 1037 | (17) | 1052 | (17) | 1053 | (17) | 1054 | (17) | 1206 | (17) | 1248 | (17) | 1252 | (17) | 1288 | (17) |
| 1292 | (17) | 146  | (18) | 306  | (18) | 398  | (18) | 669  | (18) | 681  | (18) | 708  | (18) | 767  | (18) |
| 842  | (18) | 878  | (18) | 955  | (18) | 988  | (18) | 1000 | (18) | 1064 | (18) | 1190 | (18) | 1241 | (18) |
| 1269 | (18) | 1300 | (18) | 1307 | (18) | 1372 | (18) | 1381 | (18) | 1390 | (18) | 1429 | (18) | 1551 | (18) |
| 1610 | (18) | 1626 | (18) | 1639 | (18) | 1648 | (18) | 1708 | (18) | 1711 | (18) | 1738 | (18) | 1782 | (18) |
| 16   | (19) | 229  | (19) | 253  | (19) | 256  | (19) | 334  | (19) | 346  | (19) | 502  | (19) | 535  | (19) |
| 687  | (19) | 690  | (19) | 747  | (19) | 761  | (19) | 816  | (19) | 925  | (19) | 1027 | (19) | 16   | (20) |
| 206  | (20) | 254  | (20) | 317  | (20) | 339  | (20) | 374  | (20) | 397  | (20) | 401  | (20) | 429  | (20) |
| 457  | (20) | 592  | (20) | 602  | (20) | 638  | (20) | 15   | (21) | 66   | (21) | 77   | (21) | 216  | (21) |
| 248  | (21) | 288  | (21) | 337  | (21) | 364  | (21) | 378  | (21) | 393  | (21) | 404  | (21) | 415  | (21) |
| 424  | (21) | 438  | (21) | 461  | (21) | 471  | (21) | 492  | (21) | 659  | (21) | 760  | (21) | 777  | (21) |
| 798  | (21) | 816  | (21) | 834  | (21) | 852  | (21) | 870  | (21) | 890  | (21) | 1108 | (21) | 1128 | (21) |
| 1142 | (21) | 1157 | (21) | 1319 | (21) | 82   | (22) | 99   | (22) | 159  | (23) | 175  | (23) | 297  | (23) |
| 25   | (24) | 101  | (24) | 160  | (24) | 180  | (24) | 136  | (25) | 176  | (25) | 177  | (25) | 182  | (25) |
| 183  | (25) | 184  | (25) | 185  | (25) | 186  | (25) | 187  | (25) | 188  | (25) | 189  | (25) | 190  | (25) |
| 240  | (25) | 271  | (25) | 320  | (25) | 372  | (25) | 414  | (25) | 427  | (25) | 448  | (25) | 461  | (25) |
| 479  | (25) | 11   | (26) | 100  | (26) | 166  | (26) | 394  | (26) | 579  | (26) | 709  | (26) | 727  | (26) |
| 867  | (26) | 1144 | (26) | 1244 | (26) | 1257 | (26) | 162  | (27) | 164  | (27) | 165  | (27) | 195  | (27) |
| 204  | (27) | 240  | (27) | 248  | (27) | 288  | (27) | 303  | (27) | 305  | (27) | 324  | (27) | 331  | (27) |
| 331  | (27) | 341  | (27) | 17   | (28) | 35   | (28) | 38   | (28) | 39   | (28) | 55   | (28) | 56   | (28) |
| 103  | (28) | 152  | (28) | 197  | (28) | 223  | (28) | 287  | (28) | 325  | (28) | 327  | (28) | 410  | (28) |
| 464  | (28) | 600  | (28) | 708  | (28) | 723  | (28) | 837  | (28) | 852  | (28) | 867  | (28) | 867  | (28) |
| 869  | (28) | 891  | (28) | 993  | (28) | 1013 | (28) | 1059 | (28) | 1069 | (28) | 1116 | (28) | 156  | (29) |
| 162  | (29) | 170  | (29) | 211  | (29) | 215  | (29) | 218  | (29) | 218  | (29) | 258  | (29) | 167  | (30) |
| 243  | (30) | 268  | (30) | 420  | (30) | 577  | (30) | 122  | (31) | 11   | (33) | 160  | (33) | 219  | (32) |
| 103  | (36) | 160  | (36) | 163  | (36) | 167  | (36) | 173  | (36) | 180  | (36) | 193  | (36) | 219  | (36) |
| 231  | (36) | 232  | (36) | 232  | (36) | 326  | (36) | 326  | (36) | 327  | (36) | 328  | (36) | 339  | (36) |
| 366  | (36) | 393  | (36) | 410  | (36) | 416  | (36) | 428  | (36) | 434  | (36) | 441  | (36) | 445  | (36) |
| 445  | (36) | 452  | (36) | 516  | (36) | 541  | (36) | 62   | (37) | 63   | (37) | 111  | (37) | 111  | (37) |
| 137  | (37) | 228  | (37) | 231  | (37) | 246  | (37) | 318  | (37) | 408  | (37) | 445  | (37) | 445  | (37) |
| 502  | (37) | 503  | (37) | 656  | (37) | 725  | (37) | 725  | (37) | 770  | (37) | 839  | (37) | 840  | (37) |
| 840  | (37) | 845  | (37) | 846  | (37) | 973  | (37) | 1046 | (37) | 1119 | (37) | 1119 | (37) | 1119 | (37) |
| 1120 | (37) | 1251 | (37) | 1277 |      |      |      |      |      |      |      |      |      |      |      |

|                   |            |            |            |            |            |           |            |            |
|-------------------|------------|------------|------------|------------|------------|-----------|------------|------------|
|                   | 1695 (37)  | 1696 (37)  | 1792 (37)  | 1793 (37)  | 1795 (37)  | 1923 (37) | 1925 (37)  | 1926 (37)  |
|                   | 1968 (37)  | 1970 (37)  | 1991 (37)  | 2018 (37)  | 2058 (37)  | 37 (38)   | 41 (38)    | 71 (38)    |
|                   | 72 (38)    | 73 (38)    | 74 (38)    | 124 (38)   | 128 (38)   | 129 (38)  | 130 (38)   | 132 (38)   |
|                   | 155 (38)   | 156 (38)   | 205 (38)   | 215 (38)   | 243 (38)   | 246 (38)  | 256 (38)   | 267 (38)   |
|                   | 273 (38)   | 277 (38)   | 301 (38)   | 307 (38)   | 317 (38)   | 336 (38)  | 400 (38)   | 461 (38)   |
|                   | 544 (38)   | 608 (38)   | 11 (39)    | 176 (39)   | 185 (39)   | 19 (40)   | 201 (40)   | 297 (40)   |
|                   | 347 (40)   | 371 (40)   | 412 (40)   | 414 (40)   | 544 (40)   | 563 (40)  | 574 (40)   | 698 (40)   |
|                   | 912 (40)   | 952 (40)   | 986 (40)   | 994 (40)   | 1083 (40)  | 1178 (40) | 1258 (40)  | 1260 (40)  |
|                   | 1348 (40)  | 1360 (40)  | 1464 (40)  | 1574 (40)  | 13 (41)    | 348 (41)  | 436 (41)   | 522 (41)   |
|                   | 714 (41)   | 818 (41)   | 41 (42)    | 56 (42)    | 58 (42)    | 60 (42)   | 62 (42)    |            |
| Boolean           | 199 (25)   |            |            |            |            |           |            |            |
| BOR               | 95 (6)     | 642 (21)   | 316 (26)   | 432 (28)   | 89 (30)    | 479 (30)  |            |            |
| bOSType           | 759 (6)    | 760 (6)    | 765 (6)    | 954 (16)   | 968 (16)   | 1007 (16) | 1498 (17)  | 1499 (17)  |
|                   | 1915 (18)  | 1036 (19)  | 105* (25)  | 890* (26)  | 1002 (26)  |           |            |            |
| bothClassAndProc  | 394* (26)  |            |            |            |            |           |            |            |
| botRight          | 1153 (10)  |            |            |            |            |           |            |            |
| botRight          | 170 (6)    | 2231- (6)  | 2231 (6)   | 2406- (6)  | 2406 (6)   | 2475- (6) | 2475 (6)   | 1775 (10)  |
|                   | 604- (14)  | 879 (14)   | 956 (14)   | 3136 (16)  | 3136 (16)  | 485 (18)  | 1139- (18) | 1183- (18) |
|                   | 1617- (18) | 64 (19)    | 139 (19)   | 313 (19)   | 320 (19)   | 357 (19)  | 638 (19)   | 877- (19)  |
|                   | 958 (19)   | 959 (19)   | 960- (19)  | 169 (20)   | 197 (20)   | 444 (20)  | 446 (20)   | 239 (21)   |
|                   | 267 (21)   | 124 (23)   | 259 (23)   | 259 (23)   | 167* (25)  | 471 (26)  | 1225 (26)  | 1297 (26)  |
|                   | 417 (37)   | 423 (37)   | 463 (37)   | 486 (37)   | 488 (37)   | 728 (37)  | 851 (37)   | 949 (37)   |
|                   | 949 (37)   | 981 (37)   | 1057 (37)  | 1057 (37)  | 1070 (37)  | 1070 (37) | 1856 (37)  | 1891 (37)  |
|                   | 1900 (37)  | 205 (39)   | 438- (40)  | 70* (42)   |            |           |            |            |
| botright          | 523 (10)   | 524- (10)  | 524 (10)   | 710 (10)   | 716- (10)  | 716 (10)  | 1432 (10)  |            |
| botRightCell      | 1854* (10) | 1863 (10)  | 1865 (10)  |            |            |           |            |            |
| botRightRect      | 1856* (10) | 1865 (10)  | 1866 (10)  | 1868 (10)  | 1868 (10)  | 1870 (10) | 1870 (10)  |            |
| botSlop           | 2364* (6)  | 2402 (6)   |            |            |            |           |            |            |
| bottom            | 2198 (6)   | 2215- (6)  | 2215 (6)   | 2402 (6)   | 2408 (6)   | 2409 (6)  | 2592 (6)   | 475- (10)  |
|                   | 513- (10)  | 513 (10)   | 518 (10)   | 519 (10)   | 704- (10)  | 704 (10)  | 750- (10)  | 752- (10)  |
|                   | 755 (10)   | 790- (10)  | 792- (10)  | 811 (10)   | 870- (10)  | 967- (10) | 1429- (10) | 1429 (10)  |
|                   | 1434- (10) | 1434 (10)  | 1496 (10)  | 1829- (10) | 1864 (10)  | 1867 (10) | 1868 (10)  | 2885- (10) |
|                   | 2885 (10)  | 2892- (10) | 2899- (10) | 2955- (10) | 526 (14)   | 620 (14)  | 624 (14)   | 626 (14)   |
|                   | 936- (14)  | 1032 (14)  | 499 (18)   | 500 (18)   | 509 (18)   | 511 (18)  | 513 (18)   | 1150- (18) |
|                   | 480 (19)   | 570 (19)   | 570 (19)   | 571- (19)  | 571 (19)   | 597 (19)  | 599 (19)   | 600 (19)   |
|                   | 617 (19)   | 805 (19)   | 883- (19)  | 883 (19)   | 970- (19)  | 977 (19)  | 1014 (19)  | 238 (20)   |
|                   | 264 (21)   | 783 (21)   | 785 (21)   | 976 (21)   | 166* (25)  | 680 (26)  | 680 (26)   | 731 (26)   |
|                   | 731 (26)   | 993 (26)   | 1058 (26)  | 372 (37)   | 544 (37)   | 558 (37)  | 1780- (37) | 1780 (37)  |
|                   | 383 (40)   | 449 (40)   | 451- (40)  | 534- (40)  | 1067- (40) | 1067 (40) | 1067 (40)  | 1222- (40) |
| boundRect         | 1222 (40)  | 1270- (40) | 1270 (40)  | 50* (42)   |            |           |            |            |
|                   | 1141* (10) | 1152- (10) | 1153 (10)  | 1153 (10)  | 1422* (10) | 1431 (10) | 1434 (10)  | 1435 (10)  |
|                   | 1437 (10)  | 2872* (10) | 2884- (10) | 2885 (10)  | 2885 (10)  | 2886 (10) | 2886 (10)  | 2890 (10)  |
|                   | 2891 (10)  | 2892 (10)  | 2893 (10)  |            |            |           |            |            |
| bounds            | 524 (14)   | 618 (14)   | 1216* (16) | 86* (19)   | 110 (19)   | 120 (19)  | 123 (19)   | 313 (19)   |
|                   | 314 (19)   | 357 (19)   | 358 (19)   |            |            |           |            |            |
| boundsRect        | 32* (14)   | 519 (14)   | 787* (19)  | 790- (19)  | 791 (19)   |           |            |            |
| box               | 536* (5)   | 599* (5)   | 2846* (6)  | 2848 (6)   | 3175* (6)  | 3199 (6)  | 3200 (6)   | 3200 (6)   |
|                   | 3209 (6)   | 56* (37)   | 1376* (37) | 1398 (37)  |            |           |            |            |
| bPattern          | 111* (25)  | 891* (26)  | 1008 (26)  |            |            |           |            |            |
| bPoint            | 949 (16)   | 981 (16)   | 986 (16)   | 305 (20)   | 513 (21)   | 44 (24)   | 45 (24)    | 99* (25)   |
|                   | 886* (26)  | 980 (26)   | 889 (37)   | 900 (37)   |            |           |            |            |
| bPointer          | 97* (25)   | 881* (26)  | 970 (26)   |            |            |           |            |            |
| bQDRect           | 568* (10)  |            |            |            |            |           |            |            |
| break             | 233* (7)   | 268* (8)   | 272 (8)    |            |            |           |            |            |
| BreakFollowing    | 2355* (13) | 710 (18)   | 24* (24)   | 218* (36)  | 407* (37)  | 428- (37) |            |            |
| bRect             | 397* (10)  | 415 (10)   | 418 (10)   | 419 (10)   | 491* (10)  | 523 (10)  | 524 (10)   | 547* (10)  |
|                   | 566* (10)  | 690* (10)  | 715 (10)   | 716 (10)   | 988 (16)   | 989 (16)  | 990 (16)   | 1009 (16)  |
|                   | 1033 (19)  | 1034 (19)  | 514 (21)   | 100* (25)  | 887* (26)  | 986 (26)  | 880 (37)   | 881 (37)   |
|                   | 882 (37)   | 883 (37)   | 1642 (40)  |            |            |           |            |            |
| bResType          | 125* (25)  |            |            |            |            |           |            |            |
| bRGBColor         | 114* (25)  | 892* (26)  | 1017 (26)  | 1103 (26)  |            |           |            |            |
| bRGBColor         | 996 (16)   |            |            |            |            |           |            |            |
| bRgnHandle        | 925 (16)   | 113* (25)  | 882* (26)  | 972 (26)   |            |           |            |            |
| BROTR             | 596 (17)   |            |            |            |            |           |            |            |
| BSL               | 160 (12)   | 456 (12)   | 576 (12)   | 1070 (21)  | 1368 (21)  | 89 (30)   |            |            |
| BSPoolLoop        | 1150 (37)  | 1189 (37)  | 1479 (37)  |            |            |           |            |            |
| BSR               | 1270 (6)   | 166 (8)    | 777 (10)   | 778 (10)   | 696 (12)   | 754 (12)  | 1168 (16)  | 1534 (16)  |
|                   | 1669 (16)  | 2585 (16)  | 1117 (21)  | 1134 (21)  | 1149 (21)  | 490 (26)  | 110 (30)   | 122 (30)   |
| bString           | 633 (12)   | 930 (16)   | 1004 (16)  | 1008 (16)  | 1497 (17)  | 94* (25)  | 877* (26)  | 961 (26)   |
|                   | 893 (37)   |            |            |            |            |           |            |            |
| bStringHandle     | 60 (2)     | 61 (2)     | 122* (25)  | 897* (26)  | 1061 (26)  |           |            |            |
| bStyle            | 994 (16)   | 117* (25)  | 893* (26)  | 1031 (26)  | 1101 (26)  | 130 (39)  | 147 (39)   |            |
| bTEHandle         | 108* (25)  | 884* (26)  | 974 (26)   | 1639 (40)  | 1641 (40)  | 167 (41)  |            |            |
| bTitle            | 992 (16)   | 115* (25)  | 1099 (26)  | 879 (37)   |            |           |            |            |
| btPtr             | 823* (6)   | 830- (6)   | 834 (6)    |            |            |           |            |            |
| BTST              | 94 (6)     | 96 (6)     | 98 (6)     | 100 (6)    | 121 (6)    | 123 (6)   | 125 (6)    | 2071 (16)  |
|                   | 315 (26)   | 317 (26)   | 319 (26)   | 321 (26)   | 792 (26)   | 794 (26)  | 796 (26)   |            |
| BTst              | 1398 (16)  | 1399 (16)  | 1400 (16)  | 1401 (16)  | 1402 (16)  | 1403 (16) | 921 (28)   | 1076 (28)  |
| BuildAllReserves  | 284* (27)  | 195* (28)  | 289 (28)   | 377 (28)   | 839 (28)   | 1002 (28) |            |            |
| BuildCodeReserve  | 55* (28)   | 206 (28)   | 223* (28)  | 658 (28)   | 683 (28)   | 879 (28)  |            |            |
| BuildMessage      | 183* (7)   | 1967 (16)  | 2266 (16)  |            |            |           |            |            |
| BuildObjNameTable | 427 (14)   | 126* (31)  | 31* (32)   |            |            |           |            |            |
| BusyActivate      | 74* (3)    | 70* (4)    | 146 (14)   | 215 (16)   | 1150 (16)  | 1179 (16) | 1965 (16)  | 2008 (16)  |
|                   | 2013 (16)  | 2520 (16)  |            |            |            |           |            |            |
| BusyDelay         | 79* (3)    | 88* (4)    | 158 (4)    |            |            |           |            |            |
| BusyInstall       | 94* (3)    | 109* (4)   | 489 (14)   |            |            |           |            |            |
| BusyRemove        | 100* (3)   | 166* (4)   | 98 (14)    |            |            |           |            |            |
| BusyReset         | 60* (4)    | 73 (4)     | 100 (4)    | 201* (4)   | 274 (4)    |           |            |            |
| BusyTurnOff       | 228* (4)   | 293 (4)    | 320 (4)    |            |            |           |            |            |
| ButtonTemplate    | 133 (5)    | 134* (5)   | 809 (6)    | 831 (6)    |            |           |            |            |
| ButtonTemplatePtr | 133* (5)   | 806 (6)    | 823 (6)    | 830 (6)    |            |           |            |            |
| buzzItem          | 579* (30)  | 582- (30)  | 584- (30)  | 586 (30)   |            |           |            |            |
| bVCoordinate      | 118* (25)  | 894* (26)  | 1043 (26)  |            |            |           |            |            |
| bVPoint           | 951 (16)   | 967 (16)   | 980 (16)   | 1910 (18)  | 1911 (18)  | 300 (20)  | 301 (20)   | 302 (20)   |
|                   | 46 (24)    | 119* (25)  | 895* (26)  | 1045 (26)  | 901 (37)   |           |            |            |
| bVRect            | 1010 (16)  | 1011 (16)  | 306 (20)   | 120* (25)  | 896* (26)  | 1051 (26) | 885 (37)   | 902 (37)   |
| bWindowPtr        | 982 (16)   | 1031 (19)  | 106* (25)  | 880* (26)  | 969 (26)   |           |            |            |
| BXOR              | 2862 (16)  | 596 (17)   |            |            |            |           |            |            |
| Byte              | 1445 (13)  | 893 (21)   | 243 (29)   | 517 (30)   |            |           |            |            |

|                     |           |           |           |           |           |           |          |          |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|----------|----------|--|
| byWhom              | 2837*(13) | 1266*(14) | 1274(14)  | 1281(14)  |           |           |          |          |  |
| bzCantDraw          | 334*(13)  |           |           |           |           |           |          |          |  |
| bzCantUndo          | 342*(13)  | 2812(16)  |           |           |           |           |          |          |  |
| bzClosing           | 340*(13)  | 767(17)   |           |           |           |           |          |          |  |
| bzDoFirstClick      | 351*(13)  | 2904(16)  |           |           |           |           |          |          |  |
| bzDontDoFirstClick  | 352*(13)  | 2905(16)  |           |           |           |           |          |          |  |
| bzHideClip          | 331*(13)  | 849(16)   |           |           |           |           |          |          |  |
| bzMakeModal         | 349*(13)  | 2900(16)  |           |           |           |           |          |          |  |
| bzMakeModeless      | 350*(13)  | 2900(16)  |           |           |           |           |          |          |  |
| bzQuitting          | 341*(13)  | 765(17)   |           |           |           |           |          |          |  |
| bzRedo              | 333*(13)  | 2816(16)  |           |           |           |           |          |          |  |
| bzRevertAnyways     | 344*(13)  | 985(17)   |           |           |           |           |          |          |  |
| bzSaveAnyways       | 343*(13)  | 1101(17)  |           |           |           |           |          |          |  |
| bzSaveAs            | 328*(13)  | 1409(17)  |           |           |           |           |          |          |  |
| bzSaveCopy          | 329*(13)  | 1410(17)  |           |           |           |           |          |          |  |
| bzShowClip          | 330*(13)  | 849(16)   |           |           |           |           |          |          |  |
| bzUndo              | 332*(13)  | 2814(16)  |           |           |           |           |          |          |  |
| bzUntitled          | 335*(13)  | 1468(17)  |           |           |           |           |          |          |  |
| -C-                 |           |           |           |           |           |           |          |          |  |
| c                   | 52*(4)    | 169*(14)  | 179(14)   | 185(14)   | 189(14)   | 193(14)   | 198(14)  | 202(14)  |  |
|                     | 202(14)   | 210(14)   | 211(14)   | 334*(26)  | 339(26)   | 341(26)   | 344(26)  | 166*(33) |  |
|                     | 176(33)   | 178(33)   | 179(33)   | 181(33)   | 184(33)   |           |          |          |  |
| c0                  | 1687*(37) |           |           |           |           |           |          |          |  |
| c1                  | 1687*(37) |           |           |           |           |           |          |          |  |
| cAboutApp           | 100*(13)  | 751(16)   | 846(16)   |           |           |           |          |          |  |
| CalcLabelRect       | 465*(5)   | 2207*(6)  | 2255(6)   | 2345(6)   |           |           |          |          |  |
| CalcMenuRect        | 467*(5)   | 2223*(6)  | 2256(6)   | 2312(6)   | 2375(6)   | 2474(6)   |          |          |  |
| CalcMenuSize        | 2195(6)   | 2271(6)   | 493(30)   |           |           |           |          |          |  |
| CalcMinSize         | 217*(9)   | 351*(10)  | 353(10)   | 1562*(13) | 2571*(13) | 570*(18)  | 626(18)  | 74*(23)  |  |
|                     | 233*(38)  | 265*(40)  |           |           |           |           |          |          |  |
| CalcPageStrips      | 2364*(13) | 719(18)   | 56*(24)   | 211*(36)  | 436*(37)  |           |          |          |  |
| CalcRealHeight      | 236*(38)  | 277(40)   | 295*(40)  | 335-(40)  | 1580(40)  |           |          |          |  |
| CalcSelLoc          | 270*(38)  | 391*(40)  | 1307(40)  |           |           |           |          |          |  |
| CalcViewPerPage     | 2367*(13) | 730(18)   | 68*(24)   | 214*(36)  | 479*(37)  |           |          |          |  |
| CallFileFilter      | 319*(16)  | 362(16)   |           |           |           |           |          |          |  |
| callingView         | 104*(5)   | 106*(5)   | 570*(6)   | 588*(6)   |           |           |          |          |  |
| CallInitHandler     | 26*(8)    | 180(8)    |           |           |           |           |          |          |  |
| CallJobDialog       | 1287*(37) | 1295(37)  |           |           |           |           |          |          |  |
| CallNotify          | 73*(28)   | 675(28)   |           |           |           |           |          |          |  |
| CallPrintDefault    | 1702*(37) | 1723(37)  |           |           |           |           |          |          |  |
| CallPrValidate      | 2065*(37) | 2081(37)  |           |           |           |           |          |          |  |
| callsToMoreMasters  | 2966*(13) | 325*(14)  | 363(14)   |           |           |           |          |          |  |
| CallStyleDialog     | 1324*(37) | 1335(37)  | 1348(37)  |           |           |           |          |          |  |
| cancel              | 168(17)   | 201(17)   |           |           |           |           |          |          |  |
| cancelItem          | 53*(5)    | 320(6)    | 359-(6)   |           |           |           |          |          |  |
| canClose            | 1932*(13) | 979*(14)  | 996(14)   | 1007(14)  | 16*(19)   | 53(19)    |          |          |  |
| CanDismiss          | 84*(5)    | 392*(6)   | 394-(6)   | 398-(6)   | 424(6)    | 462(6)    | 478(6)   | 576(6)   |  |
| canDismiss          | 2051*(13) | 145(21)   | 184-(21)  |           |           |           |          |          |  |
| canDo               | 215*(29)  | 218*(29)  | 243*(30)  | 250(30)   | 255(30)   | 268*(30)  | 275(30)  | 281(30)  |  |
| CanOpenDocument     | 816*(13)  | 329*(16)  | 342-(16)  | 359-(16)  | 362-(16)  | 364-(16)  | 1498(16) |          |  |
| CanPaste            | 2891*(13) | 73*(14)   | 962(40)   |           |           |           |          |          |  |
| canPurge            | 341*(27)  | 227*(28)  | 247(28)   | 618*(28)  | 641(28)   | 669(28)   | 675(28)  | 677(28)  |  |
|                     | 678(28)   | 679(28)   | 1059*(28) | 1083(28)  | 1085-(28) | 1090-(28) |          |          |  |
| CanReadLn           | 44(8)     | 95(8)     | 123(8)    | 427*(25)  | 11*(26)   | 14-(26)   | 16-(26)  | 73(33)   |  |
| canResize           | 1932*(13) | 978*(14)  | 996(14)   | 1007(14)  | 16*(19)   | 52(19)    |          |          |  |
| canSaveInPlace      | 1053*(17) | 1138-(17) | 1159-(17) | 1165-(17) | 1172(17)  |           |          |          |  |
| CanSelectCell       | 240*(9)   | 565*(9)   | 367*(10)  | 371-(10)  | 2496*(10) | 2498-(10) | 2946(10) | 3037(10) |  |
| CanSelectItem       | 556*(9)   | 2498(10)  | 2507*(10) | 2509-(10) |           |           |          |          |  |
| CanDeselect         | 86*(5)    | 399(6)    | 408*(6)   | 517(6)    |           |           |          |          |  |
| cap                 | 2535*(16) | 2585-(16) | 2592(16)  |           |           |           |          |          |  |
| CatchFailures       | 29(2)     | 219(2)    | 275(6)    | 322(6)    | 1141(6)   | 1411(6)   | 1457(6)  | 1634(6)  |  |
|                     | 1680(6)   | 1854(6)   | 1890(6)   | 2050(6)   | 2106(6)   | 2504(6)   | 2691(6)  | 188*(7)  |  |
|                     | 18*(8)    | 269(12)   | 854(14)   | 920(14)   | 999(14)   | 134(15)   | 226(16)  | 724(16)  |  |
|                     | 1252(16)  | 1414(16)  | 1481(16)  | 1494(16)  | 2005(16)  | 2077(16)  | 2148(16) | 2209(16) |  |
|                     | 2287(16)  | 2353(16)  | 2447(16)  | 2506(16)  | 79(17)    | 335(17)   | 669(17)  | 846(17)  |  |
|                     | 983(17)   | 1082(17)  | 1333(17)  | 1354(17)  | 1362(17)  | 206(18)   | 442(18)  | 61(19)   |  |
|                     | 115(19)   | 220(20)   | 640(21)   | 1224(21)  | 1258(21)  | 209(23)   | 263(37)  | 640(37)  |  |
|                     | 1171(37)  | 1503(37)  | 1548(37)  | 1594(37)  | 1722(37)  | 2080(37)  | 2109(37) | 1109(40) |  |
|                     | 1231(40)  | 51(41)    | 381(41)   | 472(41)   | 647(41)   | 836(41)   |          |          |  |
| cbPtr               | 905*(6)   | 912-(6)   | 915(6)    | 917(6)    |           |           |          |          |  |
| cCantUndo           | 94*(13)   | 1981(16)  | 1983(16)  | 2811(16)  | 2821(16)  | 2867(16)  |          |          |  |
| cCellSelect         | 46*(9)    | 2856(10)  |           |           |           |           |          |          |  |
| cChangePrinterStyle | 37*(36)   | 2108(37)  |           |           |           |           |          |          |  |
| cClear              | 132*(13)  | 133(13)   | 563(14)   | 720(40)   | 780(40)   | 841(40)   | 974(40)  |          |  |
| cClose              | 112*(13)  | 739(16)   | 865(16)   | 205(17)   |           |           |          |          |  |
| cColumnSelect       | 48*(9)    | 3115(10)  |           |           |           |           |          |          |  |
| cCopy               | 130*(13)  | 559(14)   | 766(40)   | 840(40)   | 964(40)   | 251(41)   | 254(41)  | 286(41)  |  |
|                     | 325(41)   |           |           |           |           |           |          |          |  |
| CCrsrHandle         | 13(4)     | 62(4)     | 285(4)    |           |           |           |          |          |  |
| cCut                | 129*(13)  | 557(14)   | 765(40)   | 839(40)   | 973(40)   | 418(41)   |          |          |  |
| cDebugPrinting      | 202*(13)  | 767(16)   | 2892(16)  |           |           |           |          |          |  |
| cDebugWind          | 214*(13)  | 755(16)   | 856(16)   |           |           |           |          |          |  |
| cDoFirstClick       | 205*(13)  | 782(16)   | 2903(16)  | 2904(16)  |           |           |          |          |  |
| cEditBase           | 126*(13)  | 555(14)   | 557(14)   | 559(14)   | 561(14)   | 563(14)   | 1992(16) | 1994(16) |  |
| cEditLast           | 133*(13)  | 1992(16)  |           |           |           |           |          |          |  |
| cEditSep            | 128*(13)  |           |           |           |           |           |          |          |  |
| CellExists          | 244*(9)   | 381*(10)  | 385-(10)  | 806(10)   |           |           |          |          |  |
| CellToVRect         | 265*(9)   | 395*(10)  | 499(10)   | 522(10)   | 523(10)   | 714(10)   | 715(10)  | 1409(10) |  |
|                     | 1431(10)  | 1432(10)  | 1458(10)  | 1862(10)  | 1865(10)  | 2949(10)  |          |          |  |
| Center              | 2012*(13) | 154(19)   | 346*(19)  |           |           |           |          |          |  |
| cExperimenting      | 193*(13)  | 763(16)   | 2890(16)  |           |           |           |          |          |  |
| cFinderNew          | 145*(13)  | 1482(16)  |           |           |           |           |          |          |  |
| cFinderOpen         | 149*(13)  | 1490(16)  | 1508(16)  |           |           |           |          |          |  |
| cFinderPrint        | 147*(13)  | 1452(16)  | 1488(16)  | 2449(16)  | 2461(16)  |           |          |          |  |
| CGrafPort           | 1837(18)  |           |           |           |           |           |          |          |  |
| CGrafPtr            | 584(14)   |           |           |           |           |           |          |          |  |
| ch                  | 92*(5)    | 95*(5)    | 584*(5)   | 474*(6)   | 476(6)    | 486(6)    | 493*(6)  | 497(6)   |  |
|                     | 499(6)    | 499(6)    | 502(6)    | 3018*(6)  | 3024(6)   | 3024(6)   | 3026(6)  | 3032(6)  |  |

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                     | 633*(13)  | 637*(13)  | 943*(13)  | 201*(15)  | 204 (15)  | 233*(15)  | 236 (15)  | 670*(16)  |
|                     | 676 (16)  | 1527*(16) | 1533*(16) | 1537 (16) | 1539 (16) | 2534*(16) | 2584*(16) | 2587 (16) |
|                     | 2587 (16) | 2587 (16) | 2588 (16) | 2592 (16) | 366*(25)  | 1112*(26) | 1115 (26) | 1115 (26) |
|                     | 1116 (26) | 1118 (26) | 178*(38)  | 189*(38)  | 695*(40)  | 705 (40)  | 706 (40)  | 710 (40)  |
|                     | 715 (40)  | 725 (40)  | 728 (40)  | 742 (40)  | 747 (40)  | 812*(40)  | 819 (40)  |           |
| changeToWatch       | 20*( 4)   | 120*( 4)  | 206 ( 4)  | 237 ( 4)  |           |           |           |           |
| ChangeWrap          | 273*(38)  | 347*(40)  | 1236 (40) |           |           |           |           |           |
| CHAR                | 92 ( 5)   | 584 ( 5)  | 474 ( 6)  | 3018 ( 6) | 637 (13)  | 943 (13)  | 670 (16)  | 1527 (16) |
|                     | 366 (25)  | 366 (25)  | 878 (26)  | 1112 (26) | 1112 (26) | 435 (30)  | 1687 (37) | 548 (38)  |
|                     | 562 (38)  | 677 (41)  | 711 (41)  | 813 (41)  |           |           |           |           |
| Char                | 95 ( 5)   | 493 ( 6)  | 633 (13)  | 201 (15)  | 233 (15)  | 38 (32)   | 178 (38)  | 189 (38)  |
|                     | 553 (38)  | 571 (38)  | 40 (39)   | 695 (40)  | 812 (40)  | 632 (41)  |           |           |
| Char2Pixel          | 499 (40)  |           |           |           |           |           |           |           |
| CharByte            | 1080 (14) | 720 (41)  | 794 (41)  |           |           |           |           |           |
| charCodeMask        | 1533 (16) | 2584 (16) |           |           |           |           |           |           |
| charCount           | 403*(40)  | 421*(40)  | 429 (40)  | 429 (40)  | 430 (40)  | 474 (40)  |           |           |
| CharsHandle         | 307 (40)  | 398 (40)  | 424 (40)  | 730 (41)  |           |           |           |           |
| CharWidth           | 2386 ( 6) | 2389 ( 6) |           |           |           |           |           |           |
| chBackspace         | 2935 ( 6) | 65*(25)   | 728 (40)  | 842 (41)  |           |           |           |           |
| chClear             | 68*(25)   | 715 (40)  |           |           |           |           |           |           |
| chDown-             | 2935 ( 6) | 72*(25)   | 706 (40)  |           |           |           |           |           |
| cHeadings           | 46*(36)   |           |           |           |           |           |           |           |
| CheckAdornment      | 923*(26)  | 1070 (26) | 1073 (26) | 1074 (26) | 1075 (26) | 1076 (26) | 1079 (26) | 1080 (26) |
|                     | 1081 (26) |           |           |           |           |           |           |           |
| checkBoxProc        | 866 ( 6)  | 889 ( 6)  |           |           |           |           |           |           |
| checkBoxTemplate    | 162 ( 5)  | 163*( 5)  | 891 ( 6)  | 913 ( 6)  |           |           |           |           |
| checkBoxTemplatePtr | 162*( 5)  | 888 ( 6)  | 905 ( 6)  | 912 ( 6)  |           |           |           |           |
| CheckButton         | 50*(37)   | 1415 (37) |           |           |           |           |           |           |
| checkClipboard      | 1049*(13) | 2518*(16) | 2521 (16) |           |           |           |           |           |
| CheckDeskScrap      | 1016*(13) | 375*(16)  | 2378 (16) | 2522 (16) |           |           |           |           |
| CheckDiskFile       | 1379*(13) | 156*(17)  | 985 (17)  | 1101 (17) |           |           |           |           |
| checkIt             | 218*(29)  | 268*(30)  | 275 (30)  | 285 (30)  |           |           |           |           |
| CheckItem           | 285 (30)  | 457 (30)  | 460 (30)  |           |           |           |           |           |
| checkMark           | 2469 ( 6) |           |           |           |           |           |           |           |
| CheckPrinter        | 2371*(13) | 739 (18)  | 80*(24)   | 361*(36)  | 496*(37)  | 664 (37)  | 670 (37)  | 1337 (37) |
|                     | 1929 (37) | 2140 (37) | 2168 (37) | 2180 (37) |           |           |           |           |
| CheckPrintFailure   | 1579*(37) | 1596 (37) | 1604 (37) | 1607 (37) |           |           |           |           |
| CheckReserve        | 1289 (16) | 288*(27)  | 287*(28)  | 290*(28)  | 529 (28)  |           |           |           |
| CheckRsrcUsage      | 1132 (16) | 2722 (16) | 295*(27)  | 299*(28)  | 1120 (28) |           |           |           |
| CheckScrapContents  | 2570*(13) | 153*(23)  | 238 (23)  |           |           |           |           |           |
| CheckStyleItem      | 910*(26)  | 1034 (26) | 1035 (26) | 1036 (26) | 1037 (26) | 1038 (26) | 1039 (26) | 1040 (26) |
| CheckSubview        | 543*( 6)  | 549 ( 6)  | 553 ( 6)  |           |           |           |           |           |
| checkType           | 1391*(13) | 242*(17)  | 250 (17)  |           |           |           |           |           |
| chEnter             | 499 ( 6)  | 64*(25)   | 725 (40)  |           |           |           |           |           |
| ChkPrintErr         | 111*(37)  | 235 (37)  | 1157 (37) | 1175 (37) | 1183 (37) | 1296 (37) |           |           |
| chLeft              | 2935 ( 6) | 69*(25)   | 705 (40)  |           |           |           |           |           |
| ChooseDocument      | 833*(13)  | 408*(16)  | 454*(16)  | 735 (16)  |           |           |           |           |
| ChooserByte         | 140*(37)  | 144 (37)  | 146 (37)  |           |           |           |           |           |
| ChooseSpoolFile     | 300*(36)  | 588*(37)  | 1482 (37) |           |           |           |           |           |
| CHR                 | 35 ( 5)   | 36 ( 5)   | 2395 ( 6) | 2469 ( 6) | 3165 ( 6) | 1533 (16) | 2588 (16) | 63 (25)   |
|                     | 64 (25)   | 65 (25)   | 66 (25)   | 67 (25)   | 68 (25)   | 69 (25)   | 70 (25)   | 71 (25)   |
|                     | 72 (25)   | 408 (26)  | 486 (26)  | 1116 (26) | 1135 (26) | 456 (30)  |           |           |
|                     | 69 (32)   |           |           |           |           |           |           |           |
| Chr                 | 35*( 5)   | 45 ( 5)   |           |           |           |           |           |           |
| Chr00               | 36*( 5)   | 45 ( 5)   |           |           |           |           |           |           |
| ChrIF               | 499 ( 6)  | 63*(25)   | 307 (40)  | 430 (40)  |           |           |           |           |
| chReturn            | 2935 ( 6) | 70*(25)   |           |           |           |           |           |           |
| chRight             | 67*(25)   |           |           |           |           |           |           |           |
| chSpace             | 497 ( 6)  | 66*(25)   |           |           |           |           |           |           |
| chTab               | 849*(10)  | 877*(10)  | 946*(10)  | 974*(10)  |           |           |           |           |
| chunkFound          | 2935 ( 6) | 71*(25)   |           |           |           |           |           |           |
| chUp                | 1553 ( 6) | 1567 ( 6) |           |           |           |           |           |           |
| cIconHandle         | 191*(13)  | 773 (16)  | 2895 (16) |           |           |           |           |           |
| cIdentifySoftware   | 44*(36)   |           |           |           |           |           |           |           |
| cIncludeFrame       | 1249*(13) | 1266*(13) | 517*(17)  | 524 (17)  | 534 (17)  | 536 (17)  | 539 (17)  | 541 (17)  |
| cInfo               | 546 (17)  | 547 (17)  | 556 (17)  | 617*(17)  | 645 (17)  | 645 (17)  | 656 (17)  | 665 (17)  |
|                     | 665 (17)  | 666 (17)  | 667 (17)  | 674 (17)  | 676 (17)  | 689 (17)  | 696 (17)  | 697 (17)  |
|                     | 702 (17)  | 702 (17)  | 1251*(17) | 1257 (17) | 1269 (17) | 1270 (17) | 1272 (17) | 1291*(17) |
|                     | 1345 (17) | 1349 (17) | 1350 (17) | 1352 (17) |           |           |           |           |
| cInfoPBRec          | 1249 (13) | 1266 (13) | 517 (17)  | 617 (17)  | 1251 (17) | 1291 (17) |           |           |
| cIntenseDebugging   | 197*(13)  | 771 (16)  | 2894 (16) |           |           |           |           |           |
| cl                  | 147*( 8)  | 159 ( 8)  | 160 ( 8)  | 161 ( 8)  |           |           |           |           |
| claimClipboard      | 1023*(13) | 397 (16)  | 468*(16)  | 2511 (16) | 397 (41)  |           |           |           |
| className           | 434*(25)  | 389*(26)  | 414*(26)  | 441 (26)  | 443*(26)  |           |           |           |
| CleanupMacApp       | 87*(14)   | 235 (14)  | 456 (14)  | 2727 (16) |           |           |           |           |
| clickCount          | 581*(16)  | 584*(16)  | 593*(16)  | 599 (16)  | 600 (16)  |           |           |           |
| clikLoop            | 621*( 6)  | 53 (40)   | 109 (40)  |           |           |           |           |           |
| ClipFurtherTo       | 1586*(13) | 579*(18)  | 1047 (18) | 224 (39)  |           |           |           |           |
| clipHere            | 348*(41)  |           |           |           |           |           |           |           |
| ClipRect            | 3152 (16) | 1055 (18) | 1854 (18) | 487 (19)  | 522 (19)  | 910 (21)  | 921 (37)  | 962 (37)  |
|                     | 1001 (40) | 1017 (40) |           |           |           |           |           |           |
| clipRgn             | 22 (18)   | 148 (18)  | 164 (18)  | 588 (18)  | 1025 (18) | 84 (39)   | 89 (39)   |           |
| clipStyle           | 350*(41)  | 365 (41)  | 377 (41)  |           |           |           |           |           |
| clipTEView          | 347*(41)  | 359 (41)  | 371 (41)  | 372 (41)  | 374 (41)  | 379 (41)  | 386 (41)  | 390 (41)  |
|                     | 392 (41)  | 395 (41)  | 397 (41)  | 405 (41)  |           |           |           |           |
| clipView            | 1023*(13) | 1052*(13) | 468*(16)  | 472 (16)  | 476 (16)  | 483 (16)  | 2746*(16) | 2765 (16) |
|                     | 2766 (16) | 2767 (16) | 2768 (16) | 2771 (16) | 2780 (16) |           |           |           |
| Clone               | 69*(31)   | 226*(33)  | 228*(33)  |           |           |           |           |           |
| Close               | 88*( 5)   | 420*( 6)  | 887*(13)  | 1363*(13) | 1546*(13) | 1951*(13) | 499*(16)  | 517 (16)  |
|                     | 726 (16)  | 744 (16)  | 2716 (16) | 181*(17)  | 190 (17)  | 601*(18)  | 608 (18)  | 376*(19)  |
|                     | 379 (19)  | 413 (19)  | 421 (19)  | 423 (19)  |           |           |           |           |
| CloseADocument      | 514*(16)  | 529 (16)  |           |           |           |           |           |           |
| CloseAWindow        | 188*(17)  | 212 (17)  |           |           |           |           |           |           |
| CloseByUser         | 1953*(13) | 550 (16)  | 396*(19)  |           |           |           |           |           |
| CloseCPort          | 1892 (18) |           |           |           |           |           |           |           |
| CloseDeskAcc        | 545 (16)  | 2040 (16) |           |           |           |           |           |           |
| CloseFile           | 478 (17)  | 639 (17)  | 711 (17)  | 813 (17)  | 891 (17)  | 380*(25)  | 25*(26)   | 43*(26)   |
| ClosePicture        | 1859 (18) |           |           |           |           |           |           |           |
| ClosePort           | 1894 (18) |           |           |           |           |           |           |           |

|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| ClosePrintShop      | 274*(36)  | 604*(37)  | 633 (37)  | 644 (37)  |           |           |           |           |  |
| CloseResFile        | 38 (26)   |           |           |           |           |           |           |           |  |
| CloseRgn            | 448 (18)  |           |           |           |           |           |           |           |  |
| closesDocument      | 1874*(13) | 1897*(13) | 136 (19)  | 194-(19)  |           |           |           |           |  |
| CloseSubView        | 606*(18)  | 612 (18)  |           |           |           |           |           |           |  |
| CloseWindow         | 279 (14)  |           |           |           |           |           |           |           |  |
| CloseWmgrWindow     | 962*(13)  | 525 (16)  | 540*(16)  | 740 (16)  | 1628 (16) |           |           |           |  |
| clPtr               | 1182*( 6) | 1189-( 6) | 1193 ( 6) |           |           |           |           |           |  |
| ClrAppFiles         | 1500 (16) |           |           |           |           |           |           |           |  |
| ClusterTemplate     | 242 ( 5)  | 243*( 5)  | 1168 ( 6) | 1190 ( 6) |           |           |           |           |  |
| ClusterTemplatePtr  | 242*( 5)  | 1165 ( 6) | 1182 ( 6) | 1189 ( 6) |           |           |           |           |  |
| cMargins            | 45*(36)   |           |           |           |           |           |           |           |  |
| cmd                 | 1438*(16) | 1452 (16) | 1488-(16) | 1490-(16) | 1498 (16) | 1954*(16) | 1967 (16) | 1979-(16) |  |
|                     | 1981 (16) | 1988 (16) | 1989 (16) | 1992 (16) | 1992 (16) | 1994 (16) | 1996 (16) | 2010 (16) |  |
| CmdArray            | 17*(30)   | 18 (30)   |           |           |           |           |           |           |  |
| cmdDone             | 951*(13)  | 2801*(16) | 2809 (16) | 2813 (16) | 2830 (16) |           |           |           |  |
| CmdFromMenuItem     | 1979 (16) | 181*(29)  | 55*(30)   | 65-(30)   | 76-(30)   | 77 (30)   | 89-(30)   |           |  |
| cmdName             | 2805*(16) | 2822-(16) | 2824 (16) | 2825 (16) |           |           |           |           |  |
| CmdNumber           | 3039 ( 6) | 646 ( 9)  | 2778 (10) | 642 (13)  | 816 (13)  | 833 (13)  | 838 (13)  | 842 (13)  |  |
|                     | 843 (13)  | 852 (13)  | 859 (13)  | 869 (13)  | 952 (13)  | 994 (13)  | 1198 (13) | 1231 (13) |  |
|                     | 1248 (13) | 1284 (13) | 1292 (13) | 1301 (13) | 1316 (13) | 1332 (13) | 1347 (13) | 1404 (13) |  |
|                     | 1702 (13) | 2393 (13) | 2432 (13) | 2461 (13) | 2801 (13) | 245 (15)  | 329 (16)  | 408 (16)  |  |
|                     | 684 (16)  | 701 (16)  | 1438 (16) | 1883 (16) | 1883 (16) | 1954 (16) | 2130 (16) | 2183 (16) |  |
|                     | 2801 (16) | 2849 (16) | 2926 (16) | 135 (17)  | 299 (17)  | 515 (17)  | 734 (17)  | 784 (17)  |  |
|                     | 901 (17)  | 1036 (17) | 1206 (17) | 1247 (17) | 1287 (17) | 1399 (17) | 784 (18)  | 12 (22)   |  |
|                     | 160 (24)  | 150*(29)  | 181 (29)  | 193 (29)  | 207 (29)  | 215 (29)  | 218 (29)  | 243 (29)  |  |
|                     | 248 (29)  | 253 (29)  | 257 (29)  | 265 (29)  | 55 (30)   | 100 (30)  | 148 (30)  | 243 (30)  |  |
|                     | 268 (30)  | 517 (30)  | 539 (30)  | 561 (30)  | 576 (30)  | 596 (30)  | 339 (36)  | 401 (36)  |  |
|                     | 654 (37)  | 1427 (37) | 103 (38)  | 182 (38)  | 185 (38)  | 192 (38)  | 400 (38)  | 466 (38)  |  |
|                     | 512 (38)  | 757 (40)  | 795 (40)  | 828 (40)  | 13 (41)   | 320 (41)  | 520 (41)  |           |  |
| CmdTable            | 19*(30)   | 25 (30)   | 323 (30)  |           |           |           |           |           |  |
| CmdToMenuItem       | 193*(29)  | 100*(30)  | 128 (30)  | 156 (30)  | 252 (30)  | 277 (30)  | 526 (30)  | 548 (30)  |  |
|                     | 605 (30)  |           |           |           |           |           |           |           |  |
| CmdToName           | 189 (14)  | 2824 (16) | 207*(29)  | 148*(30)  |           |           |           |           |  |
| cModalToggle        | 211*(13)  | 780 (16)  | 2899 (16) | 2900 (16) |           |           |           |           |  |
| cMouseCommand       | 138*(13)  |           |           |           |           |           |           |           |  |
| cNew                | 104*(13)  | 730 (16)  | 852 (16)  |           |           |           |           |           |  |
| cNewInspectorWindow | 210*(13)  | 794 (16)  | 860 (16)  |           |           |           |           |           |  |
| cNewLast            | 106*(13)  | 730 (16)  |           |           |           |           |           |           |  |
| cNoCommand          | 92*(13)   | 669 (14)  | 2821 (16) | 2871 (16) |           |           |           |           |  |
| cnPtr               | 166*(21)  | 174-(21)  | 177 (21)  |           |           |           |           |           |  |
| CntlAdornment       | 1596 (13) | 2067 (13) | 394 (18)  | 141 (21)  | 154*(25)  | 898 (26)  | 923 (26)  |           |  |
| CntlFrame           | 396*(18)  | 493 (18)  |           |           |           |           |           |           |  |
| codeRes             | 726*(28)  | 730 (28)  | 734 (28)  |           |           |           |           |           |  |
| codeReserve         | 304*(27)  | 326*(28)  | 353-(28)  | 355-(28)  | 374 (28)  | 379-(28)  |           |           |  |
| codeShort           | 304*(27)  | 326*(28)  | 380-(28)  |           |           |           |           |           |  |
| codeSize            | 619*(28)  | 641-(28)  | 654 (28)  |           |           |           |           |           |  |
| codeVal             | 403*(28)  | 496 (28)  |           |           |           |           |           |           |  |
| colInset            | 98*( 9)   | 182*( 9)  | 437*( 9)  | 530*( 9)  | 40*(10)   | 62 (10)   | 63 (10)   | 65 (10)   |  |
|                     | 162 (10)  | 163 (10)  | 165 (10)  | 264-(10)  | 2209*(10) | 2219 (10) | 2449*(10) | 2458 (10) |  |
| colorRect           | 2371*( 6) | 2379 ( 6) | 2400 ( 6) |           |           |           |           |           |  |
| ColToVRect          | 269*( 9)  | 415 (10)  | 432*(10)  | 876 (10)  | 1603 (10) |           |           |           |  |
| ColWidth            | 433*( 9)  |           |           |           |           |           |           |           |  |
| colWidth            | 96*( 9)   | 178*( 9)  | 36*(10)   | 127 (10)  | 225 (10)  | 260-(10)  | 262-(10)  | 696*(10)  |  |
|                     | 726-(10)  | 731 (10)  | 769*(10)  | 782-(10)  | 799 (10)  | 2205*(10) | 2218 (10) |           |  |
| command             | 971*(13)  | 2244*(16) | 2263 (16) | 2266 (16) | 2271 (16) | 2273 (16) | 2277 (16) | 2279 (16) |  |
|                     | 2279 (16) | 2285 (16) | 2292 (16) | 2297 (16) | 2298 (16) | 2301 (16) | 2310 (16) | 830*(40)  |  |
| commandToPerform    | 616*(13)  | 714*(13)  | 214*(15)  | 627*(16)  | 630-(16)  | 637-(16)  | 640-(16)  | 643-(16)  |  |
|                     | 646-(16)  | 649-(16)  | 652-(16)  | 656-(16)  | 659-(16)  | 1383*(16) | 1416 (16) | 1417 (16) |  |
|                     | 1418 (16) |           |           |           |           |           |           |           |  |
| Commit              | 2473*(13) | 567 (16)  | 44*(22)   |           |           |           |           |           |  |
| CommitLastCommand   | 969*(13)  | 220 (16)  | 385 (16)  | 560*(16)  | 2282 (16) | 210 (17)  | 989 (17)  | 1108 (17) |  |
| CompactMem          | 261 (28)  |           |           |           |           |           |           |           |  |
| Compare             | 51*( 1)   | 73*( 2)   | 77-( 2)   | 79-( 2)   | 81-( 2)   | 277*(11)  | 664*(12)  | 697 (12)  |  |
| CompareKey          | 326*( 2)  | 330-( 2)  | 332-( 2)  | 334-( 2)  | 338 ( 2)  |           |           |           |  |
| compareResult       | 684*(12)  | 697-(12)  | 698 (12)  | 702 (12)  | 703 (12)  | 734*(12)  | 756-(12)  | 757 (12)  |  |
|                     | 762 (12)  |           |           |           |           |           |           |           |  |
| COMPDATE            | 1733 (16) | 323 (32)  | 1037 (37) | 1622 (40) |           |           |           |           |  |
| CompleteTyping      | 580*(38)  | 904 (40)  | 896 (41)  | 909*(41)  |           |           |           |           |  |
| COMPTIME            | 1733 (16) | 323 (32)  | 1037 (37) | 1622 (40) |           |           |           |           |  |
| ComputeAddress      | 226*(11)  | 94 (12)   | 132 (12)  | 151*(12)  | 160-(12)  | 174-(12)  | 192 (12)  | 388 (12)  |  |
|                     | 639 (12)  |           |           |           |           |           |           |           |  |
| Computesize         | 1564*(13) | 2089*(13) | 382 (18)  | 619*(18)  | 1728 (18) | 226*(21)  | 231 (21)  | 240*(38)  |  |
|                     | 366*(40)  | 378 (40)  |           |           |           |           |           |           |  |
| CONCAT              | 2349 ( 6) | 49 ( 8)   | 100 ( 8)  | 128 ( 8)  | 160 ( 8)  | 638 (12)  | 655 (12)  | 90 (15)   |  |
|                     | 597 (17)  | 1114 (18) | 131 (24)  | 344 (26)  | 540 (26)  | 560 (26)  | 915 (26)  | 917 (26)  |  |
|                     | 929 (26)  | 931 (26)  | 959 (26)  | 984 (26)  | 990 (26)  | 992 (26)  | 994 (26)  | 1014 (26) |  |
|                     | 1027 (26) | 1029 (26) | 1041 (26) | 1049 (26) | 1055 (26) | 1057 (26) | 1059 (26) | 1082 (26) |  |
|                     | 778 (28)  | 334 (30)  | 184 (33)  | 124 (32)  | 365 (37)  |           |           |           |  |
| Concat              | 312 (28)  | 109 (32)  |           |           |           |           |           |           |  |
| condense            | 1039 (26) |           |           |           |           |           |           |           |  |
| ConfigRecord        | 170*(25)  | 197 (25)  | 234 (25)  | 74 (26)   |           |           |           |           |  |
| configuration       | 234*(25)  | 74*(26)   | 108 (26)  | 111 (26)  |           |           |           |           |  |
| constChars          | 2807*(16) | 2819 (16) | 2825 (16) | 1465*(17) | 1469 (17) | 1473 (17) |           |           |  |
| ConstrainOnce       | 3202*(16) | 3259 (16) | 3307 (16) | 3346 (16) |           |           |           |           |  |
| constTitle          | 2876*(13) | 2884*(13) | 1063*(14) | 1090-(14) | 1101-(14) | 1105-(14) | 1243*(14) | 1247 (14) |  |
|                     | 1251 (14) | 20*(19)   | 66 (19)   | 69 (19)   | 89*(19)   | 142 (19)  | 145 (19)  | 1379*(37) |  |
|                     | 1402 (37) | 1403 (37) |           |           |           |           |           |           |  |
| ContainsClipType    | 1663*(13) | 76 (14)   | 669*(18)  | 674-(18)  | 256*(38)  | 563*(40)  | 565-(40)  |           |  |
| ContainsMouse       | 1619*(13) | 2091*(13) | 681*(18)  | 688-(18)  | 1276 (18) | 1314 (18) | 82 (21)   | 248*(21)  |  |
|                     | 253-(21)  | 477 (21)  | 482 (21)  |           |           |           |           |           |  |
| continuePrinting    | 1437*(16) | 1477-(16) | 1504 (16) | 1505-(16) |           |           |           |           |  |
| ContinuousStyle     | 276*(38)  | 319 (40)  | 443 (40)  | 543*(40)  | 555-(40)  |           |           |           |  |
| contrlHilite        | 651-(21)  |           |           |           |           |           |           |           |  |
| contrlOwner         | 610-(21)  | 615-(21)  | 699 (21)  | 700-(21)  | 707-(21)  |           |           |           |  |
| contrlRect          | 1328 (21) | 1330 (21) | 1393 (21) |           |           |           |           |           |  |
| contrlTitle         | 743 (21)  |           |           |           |           |           |           |           |  |
| contrlVis           | 407 (20)  | 421-(20)  | 609-(21)  | 614-(21)  | 649-(21)  | 697 (21)  | 704-(21)  | 904 (21)  |  |
|                     | 905-(21)  | 907-(21)  | 1222-(21) | 1256-(21) | 1390 (21) |           |           |           |  |

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| ControlArea         | 805 ( 6)  | 887 ( 6)  | 1015 ( 6) | 1266 ( 6) | 1540 ( 6) | 1759 ( 6) | 1958 ( 6) | 2210 ( 6) |
|                     | 2226 ( 6) | 2305 ( 6) | 2814 ( 6) | 2890 ( 6) | 3297 ( 6) | 2093*(13) | 252 (21)  | 260*(21)  |
|                     | 278 (21)  | 356 (21)  | 970 (21)  |           |           |           |           |           |
| controlChars        | 558*( 5)  | 2960 ( 6) | 2987-( 6) |           |           |           |           |           |
| ControlCharSet      | 45*( 5)   | 558 ( 5)  | 563 ( 5)  |           |           |           |           |           |
| ControlHandle       | 2150 (13) | 12 (21)   | 628 (21)  |           |           |           |           |           |
| ControlList         | 587-(14)  |           |           |           |           |           |           |           |
| controlList         | 647-(21)  | 647 (21)  |           |           |           |           |           |           |
| ControlTemplate     | 2043 (13) | 2044*(13) | 151 (21)  | 175 (21)  |           |           |           |           |
| ControlTemplatePtr  | 2043*(13) | 138 (21)  | 166 (21)  | 174 (21)  |           |           |           |           |
| convertClipboard    | 1006*(13) | 190*(16)  | 217 (16)  | 1661*(16) | 1673-(16) | 1676 (16) | 1678 (16) |           |
| Converter           | 163*(14)  | 169 (14)  |           |           |           |           |           |           |
| coord               | 1713*(13) | 2418*(13) | 832*(18)  | 834 (18)  | 435*(20)  | 444-(20)  | 445 (20)  | 446-(20)  |
|                     | 447 (20)  | 220*(24)  | 353*(36)  | 1909*(37) | 1914 (37) | 343*(38)  | 934*(40)  | 939 (40)  |
|                     | 941 (40)  |           |           |           |           |           |           |           |
| cOpen               | 107*(13)  | 733 (16)  | 853 (16)  |           |           |           |           |           |
| cOpenLast           | 109*(13)  | 733 (16)  |           |           |           |           |           |           |
| Copy                | 594 (12)  | 2164 (16) | 592 (17)  | 109 (32)  |           |           |           |           |
| copyInfo            | 1249*(13) | 1317*(13) | 1333*(13) | 516*(17)  | 521 (17)  | 1052*(17) | 1098-(17) | 1100 (17) |
|                     | 1133 (17) | 1173 (17) | 1248*(17) | 1257 (17) | 1288*(17) | 1345 (17) |           |           |
| CopyName            | 399*(26)  | 439 (26)  | 441 (26)  |           |           |           |           |           |
| CopyRgn             | 1999 (10) | 2025 (10) | 2032 (10) | 2911 (10) | 450 (18)  | 587 (19)  |           |           |
| CopyStr255          | 834 ( 6)  | 917 ( 6)  | 1044 ( 6) | 1193 ( 6) | 2749 ( 6) | 2293 (10) | 262 (18)  | 203 (19)  |
|                     | 193 (21)  | 220*(25)  | 50*(26)   | 156 (40)  |           |           |           |           |
| count               | 237-( 8)  | 241-( 8)  | 78*( 9)   | 883-(10)  | 883 (10)  | 885 (10)  | 887 (10)  | 898-(10)  |
|                     | 898 (10)  | 899 (10)  | 980-(10)  | 980 (10)  | 982 (10)  | 984 (10)  | 995-(10)  | 995 (10)  |
|                     | 996 (10)  | 1167*(10) | 1190 (10) | 1199-(10) | 1199 (10) | 1201 (10) | 1202 (10) | 1213 (10) |
|                     | 1215-(10) | 1215 (10) | 1216 (10) | 1217 (10) | 1225-(10) | 1225 (10) | 1226 (10) | 1227 (10) |
|                     | 1251*(10) | 1274 (10) | 1283-(10) | 1283 (10) | 1285 (10) | 1286 (10) | 1297 (10) | 1299-(10) |
|                     | 1299 (10) | 1300 (10) | 1301 (10) | 1309-(10) | 1309 (10) | 1310 (10) | 1311 (10) | 1545-(10) |
|                     | 1546 (10) | 1553-(10) | 1553 (10) | 1577-(10) | 1590-(10) | 1590 (10) | 1592-(10) | 1594-(10) |
|                     | 1645-(10) | 1646 (10) | 1653-(10) | 1653 (10) | 1677-(10) | 1690-(10) | 1690 (10) | 1692-(10) |
|                     | 1694-(10) | 2099 (10) | 2100 (10) | 2122 (10) | 2123 (10) | 367*(17)  | 378-(17)  | 379 (17)  |
|                     | 411*(17)  | 423-(17)  | 424 (17)  |           |           |           |           |           |
|                     | 639 (14)  |           |           |           |           |           |           |           |
| CountAppFiles       | 976*(13)  | 577*(16)  | 600-(16)  | 1566 (16) |           |           |           |           |
| CountClicks         | 2014*(13) | 941*(19)  | 981 (19)  | 994-(19)  | 994 (19)  |           |           |           |
| counter             | 2196 ( 6) | 451 (30)  |           |           |           |           |           |           |
| CountMItems         | 404*(19)  | 418 (19)  |           |           |           |           |           |           |
| CountOpenWindows    | 113 (28)  | 441 (28)  | 484 (28)  | 491 (28)  | 807 (28)  |           |           |           |
| CountResources      | 1536*(13) | 2760 (16) | 3051 (16) | 695*(18)  | 698-(18)  | 700-(18)  | 919 (18)  |           |
| CountSubViews       | 161*(13)  | 674 (37)  | 700 (37)  |           |           |           |           |           |
| cPageSetup          | 131*(13)  | 561 (14)  | 2914 (16) | 773 (40)  | 844 (40)  | 456 (41)  |           |           |
| cPaste              | 155*(13)  | 318 (17)  | 788 (18)  |           |           |           |           |           |
| cPrFileBase         | 159*(13)  | 318 (17)  | 788 (18)  |           |           |           |           |           |
| cPrFileMax          | 163*(13)  | 662 (37)  | 699 (37)  |           |           |           |           |           |
| cPrint              | 162*(13)  | 668 (37)  | 701 (37)  |           |           |           |           |           |
| cPrintOne           | 49*(36)   |           |           |           |           |           |           |           |
| cPrintPageNums      | 166*(13)  |           |           |           |           |           |           |           |
| cPrintSpoolFile     | 164*(13)  | 1464 (37) |           |           |           |           |           |           |
| cPrintToFile        | 168*(13)  | 787 (18)  |           |           |           |           |           |           |
| cPrViewBase         | 172*(13)  | 787 (18)  |           |           |           |           |           |           |
| cPrViewMax          | 117*(13)  | 722 (16)  | 847 (16)  |           |           |           |           |           |
| cQuit               | 665 (17)  |           |           |           |           |           |           |           |
| Create              | 652*(13)  | 57*(15)   | 65-(15)   | 65 (15)   | 85-(15)   | 93-(15)   | 180 (15)  |           |
| CreateAView         | 807 ( 6)  | 889 ( 6)  | 1017 ( 6) | 2170*(13) | 542 (21)  | 623*(21)  | 971 (21)  |           |
| CreateChgrControl   | 274*( 9)  | 488*(10)  | 552 (10)  | 1995 (10) | 1998 (10) | 2905 (10) | 2908 (10) | 3000 (10) |
| CreateHighlightRgn  | 676 (17)  |           |           |           |           |           |           |           |
| CreateResFile       | 1826*(13) | 36 (20)   | 68 (20)   | 206*(20)  |           |           |           |           |
| CreateScrollBar     | 180*(13)  |           |           |           |           |           |           |           |
| cReduce50           | 181*(13)  |           |           |           |           |           |           |           |
| cReduceToFit        | 209*(13)  | 778 (16)  | 2902 (16) |           |           |           |           |           |
| cRefreshFrontWindow | 141*(13)  |           |           |           |           |           |           |           |
| cRememberStyle      | 203*(13)  | 765 (16)  | 2891 (16) |           |           |           |           |           |
| cReportEvt          | 195*(13)  | 769 (16)  | 2893 (16) |           |           |           |           |           |
| cReportMenuChoices  | 115*(13)  | 329 (17)  | 397 (17)  |           |           |           |           |           |
| cRevert             | 3205-( 6) | 352-(40)  | 354-(40)  |           |           |           |           |           |
| crOnly              | 47*( 9)   | 3082 (10) |           |           |           |           |           |           |
| cRowSelect          | 111*(13)  | 322 (17)  | 325 (17)  | 396 (17)  | 1409 (17) |           |           |           |
| cSave               | 113*(13)  | 322 (17)  | 392 (17)  | 1409 (17) |           |           |           |           |
| cSaveAs             | 114*(13)  | 322 (17)  | 326 (17)  | 393 (17)  | 1410 (17) |           |           |           |
| cSaveCopy           | 135*(13)  | 861 (40)  | 966 (40)  |           |           |           |           |           |
| cSelectAll          | 174*(13)  |           |           |           |           |           |           |           |
| cShowBorders        | 39*(36)   | 677 (37)  | 704 (37)  |           |           |           |           |           |
| cShowBreaks         | 116*(13)  | 742 (16)  | 848 (16)  | 849 (16)  |           |           |           |           |
| cShowClipboard      | 182*(13)  |           |           |           |           |           |           |           |
| cShowFullSize       | 50*(36)   |           |           |           |           |           |           |           |
| cShowPageNums       | 47*(36)   |           |           |           |           |           |           |           |
| cStartPgNum         | 140*(13)  |           |           |           |           |           |           |           |
| cStyleChange        | 48*(36)   |           |           |           |           |           |           |           |
| cTopToBottom        | 200*(13)  | 789 (16)  | 2910 (16) |           |           |           |           |           |
| cTraceIdle          | 198*(13)  | 787 (16)  | 2909 (16) |           |           |           |           |           |
| cTraceSetupMenus    | 185*(13)  | 38 (21)   |           |           |           |           |           |           |
| cTrackingControl    | 137*(13)  | 32 (40)   | 645 (41)  |           |           |           |           |           |
| cTyping             | 127*(13)  | 555 (14)  | 798 (16)  | 2828 (16) | 2885 (16) |           |           |           |
| cUndo               | 1110*(28) | 1128 (28) |           |           |           |           |           |           |
| curJTOffset         | 1042*(28) |           |           |           |           |           |           |           |
| curLimit            | 1837*(16) | 1847-(16) | 1848 (16) | 1850 (16) | 1856-(16) | 1860 (16) | 1861-(16) | 1861 (16) |
| currCohandler       | 1250*(37) | 1262-(37) | 1263 (37) | 1265-(37) | 1265 (37) |           |           |           |
| currCoord           | 433*( 5)  | 486*( 5)  | 2102 ( 6) | 2156-( 6) | 2484*( 6) | 2516 ( 6) |           |           |
| currentItem         | 751*(26)  | 754-(26)  | 755 (26)  |           |           |           |           |           |
| currentKeyScript    | 104*(18)  | 109-(18)  | 112 (18)  | 120 (18)  |           |           |           |           |
| currentPtr          | 3284*( 6) | 3290 ( 6) | 3291 ( 6) |           |           |           |           |           |
| currentText         | 600*(28)  | 602-(28)  | 605 (28)  |           |           |           |           |           |
| curResLoad          | 887*(16)  | 891-(16)  | 892 (16)  | 895 (16)  | 896 (16)  | 897-(16)  | 1028*(16) | 1032-(16) |
| currHandler         | 1033 (16) | 1036 (16) | 1037 (16) | 1039 (16) | 1042-(16) |           |           |           |
|                     | 518*(17)  | 523-(17)  | 526 (17)  |           |           |           |           |           |
| currName            | 1748*(16) | 1763 (16) | 1766 (16) | 1774-(16) | 1784 (16) | 1791 (16) | 1796 (16) |           |
| currTick            | 3115*(16) | 3138-(16) | 3163 (16) | 3164 (16) |           |           |           |           |
| currTranslation     |           |           |           |           |           |           |           |           |

|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| curSelEnd           | 817*(41)  | 833=(41)  | 846 (41)  |           |           |           |           |           |  |
| curSelStart         | 816*(41)  | 832=(41)  | 845 (41)  | 846 (41)  | 846 (41)  | 847 (41)  | 850 (41)  |           |  |
| Cursor              | 11 (4)    | 28 (4)    | 52 (4)    | 61 (4)    | 312 (4)   |           |           |           |  |
| CursorInfo          | 9*(4)     | 37 (4)    |           |           |           |           |           |           |  |
| cursorIsSet         | 3002*(16) | 3040=(16) | 3049=(16) | 3089 (16) | 3091 (16) |           |           |           |  |
| cursorRgn           | 703*(13)  | 1630*(13) | 1633*(13) | 1119*(16) | 1143 (16) | 1146 (16) | 1160 (16) | 842*(18)  |  |
|                     | 1263*(18) | 1277 (18) | 1287 (18) | 215*(38)  | 912*(40)  | 922 (40)  |           |           |  |
| cursorView          | 3005*(16) | 3046=(16) | 3047 (16) | 3051 (16) | 3054 (16) |           |           |           |  |
| cursorWindow        | 3004*(16) | 3031=(16) | 3032 (16) | 3032 (16) | 3033=(16) | 3036=(16) | 3041 (16) | 3043 (16) |  |
|                     | 3046 (16) |           |           |           |           |           |           |           |  |
| CurStackBase        | 1039*(28) | 1046 (28) |           |           |           |           |           |           |  |
| curStart            | 565*(38)  | 568*(38)  | 701*(41)  | 720 (41)  | 722=(41)  | 722 (41)  | 730 (41)  | 733 (41)  |  |
|                     | 737 (41)  | 778 (41)  | 779 (41)  | 780 (41)  | 788*(41)  | 794 (41)  |           |           |  |
| cVarClipPicSize     | 207*(13)  |           |           |           |           |           |           |           |  |
| cVertexSelect       | 49*(9)    | 3149 (10) |           |           |           |           |           |           |  |
| CyanBit             | 74*(6)    | 96 (6)    | 295*(26)  | 317 (26)  |           |           |           |           |  |
| cyanColor           | 131 (6)   | 802 (26)  |           |           |           |           |           |           |  |
| -D-                 |           |           |           |           |           |           |           |           |  |
| data                | 502*(5)   | 2718 (6)  | 2749 (6)  | 14*(16)   | 43 (16)   | 359*(30)  | 378 (30)  |           |  |
| dataBytes           | 1045*(17) | 1126=(17) | 1128 (17) | 1129 (17) |           |           |           |           |  |
| dataForkBytes       | 1238*(13) | 352*(17)  | 355=(17)  | 355 (17)  |           |           |           |           |  |
| DataHandle          | 880 (19)  |           |           |           |           |           |           |           |  |
| dataPerm            | 1193*(13) | 721*(17)  | 725 (17)  | 414*(25)  | 580*(26)  | 607 (26)  | 618 (26)  |           |  |
| dataRefNum          | 860=(17)  | 861 (17)  |           |           |           |           |           |           |  |
| dataRefnum          | 1194*(13) | 628*(17)  | 639 (17)  | 659=(17)  | 705 (17)  | 707 (17)  | 711 (17)  | 722*(17)  |  |
|                     | 726 (17)  | 801*(17)  | 813 (17)  | 843=(17)  | 851 (17)  | 869 (17)  | 877 (17)  | 878=(17)  |  |
|                     | 891 (17)  | 1221*(17) | 1237 (17) | 1238 (17) | 1316*(17) | 1337 (17) | 1339 (17) | 380*(25)  |  |
|                     | 415*(25)  | 25*(26)   | 33 (26)   | 34 (26)   | 581*(26)  | 624=(26)  | 629=(26)  |           |  |
| dataType            | 1028*(13) | 1665*(13) | 1090*(16) | 1098=(16) | 1100 (16) | 1237*(18) | 1248 (18) | 1254 (18) |  |
|                     | 259*(38)  | 834*(40)  | 846 (40)  | 1076*(40) | 1111 (40) | 1125 (40) | 441*(41)  | 474 (41)  |  |
|                     | 479 (41)  |           |           |           |           |           |           |           |  |
| debugParams         | 31*(14)   | 39 (14)   |           |           |           |           |           |           |  |
| debugParamsHndl     | 40*(14)   | 441 (14)  | 518 (14)  |           |           |           |           |           |  |
| debugParamsPtr      | 39*(14)   | 40 (14)   |           |           |           |           |           |           |  |
| dceHnd              | 2064*(16) | 2070=(16) | 2071 (16) | 2071 (16) |           |           |           |           |  |
| dCtlFlags           | 2071 (16) |           |           |           |           |           |           |           |  |
| DctlHandle          | 2064 (16) |           |           |           |           |           |           |           |  |
| Dctlhandle          | 2060 (16) |           |           |           |           |           |           |           |  |
| Dec2Num             | 3527 (6)  |           |           |           |           |           |           |           |  |
| Decimal             | 3513 (6)  |           |           |           |           |           |           |           |  |
| decNumber           | 292*(25)  | 479*(26)  | 489 (26)  | 490=(26)  | 490 (26)  |           |           |           |  |
| decRec              | 3513*(6)  | 3524 (6)  | 3525 (6)  | 3527 (6)  |           |           |           |           |  |
| defaultItem         | 52*(5)    | 319 (6)   | 358=(6)   |           |           |           |           |           |  |
| defaultName         | 1349*(13) | 1400*(17) |           |           |           |           |           |           |  |
| DefaulSize          | 231*(25)  | 59*(26)   | 369 (26)  | 377 (26)  |           |           |           |           |  |
| defClickLoopProc    | 622 (6)   | 597*(38)  | 53=(40)   | 109=(40)  |           |           |           |           |  |
| DefineConfiguration | 399 (14)  | 234*(25)  | 74*(26)   |           |           |           |           |           |  |
| Delay               | 60 (6)    |           |           |           |           |           |           |           |  |
| delayTicks          | 60*(4)    | 201*(4)   | 209 (4)   | 218 (4)   |           |           |           |           |  |
| DelColAt            | 279*(9)   | 839*(10)  | 1035 (10) | 1059 (10) | 1896 (10) |           |           |           |  |
| DelColFirst         | 284*(9)   | 1033*(10) |           |           |           |           |           |           |  |
| DelColLast          | 289*(9)   | 1057*(10) |           |           |           |           |           |           |  |
| Delete              | 289 (2)   | 312 (2)   | 212*(11)  | 183*(12)  | 1097 (14) | 1104 (14) | 1251 (14) | 609 (16)  |  |
|                     | 620 (16)  | 2047 (16) | 224 (17)  | 234 (17)  | 493 (17)  | 1465 (18) | 1480 (18) | 1539 (18) |  |
|                     | 348 (26)  |           |           |           |           |           |           |           |  |
| DeleteAll           | 217*(11)  | 219*(12)  | 352 (12)  |           |           |           |           |           |  |
| DeleteDocument      | 899*(13)  | 607*(16)  | 104 (17)  |           |           |           |           |           |  |
| DeleteFile          | 645 (17)  | 944 (17)  | 1263 (17) | 1303 (17) | 1365 (17) | 386*(25)  | 133*(26)  | 149=(26)  |  |
| DeleteFreeWindow    | 902*(13)  | 618*(16)  | 239 (19)  | 728 (19)  |           |           |           |           |  |
| DeleteMenu          | 2282 (6)  | 2325 (6)  | 2445 (6)  |           |           |           |           |           |  |
| DeleteView          | 1415*(13) | 222*(17)  | 295 (18)  |           |           |           |           |           |  |
| DeleteWindow        | 1418*(13) | 232*(17)  | 237 (19)  |           |           |           |           |           |  |
| DelItemAt           | 569*(9)   | 2520*(10) | 2535 (10) | 2548 (10) |           |           |           |           |  |
| DelItemFirst        | 570*(9)   | 2533*(10) |           |           |           |           |           |           |  |
| DelItemLast         | 571*(9)   | 2546*(10) |           |           |           |           |           |           |  |
| DelRowAt            | 280*(9)   | 936*(10)  | 1047 (10) | 1070 (10) | 1922 (10) | 2522 (10) |           |           |  |
| DelRowFirst         | 285*(9)   | 1045*(10) |           |           |           |           |           |           |  |
| DelRowLast          | 290*(9)   | 1068*(10) |           |           |           |           |           |           |  |
| delStyle            | 712*(41)  | 737 (41)  | 740 (41)  | 767 (41)  |           |           |           |           |  |
| delta               | 1568*(13) | 1570*(13) | 1817*(13) | 1831*(13) | 1833*(13) | 1843*(13) | 2246*(13) | 2573*(13) |  |
|                     | 3113*(16) | 3291 (16) | 3303 (16) | 3304 (16) | 3310 (16) | 3310 (16) | 3318 (16) | 3318 (16) |  |
|                     | 1554*(18) | 1564 (18) | 1570 (18) | 1601 (18) | 1683*(18) | 1708*(18) | 1722 (18) | 182*(20)  |  |
|                     | 192=(20)  | 196 (20)  | 198 (20)  | 254*(20)  | 262 (20)  | 263 (20)  | 263 (20)  | 264 (20)  |  |
|                     | 265 (20)  | 265 (20)  | 268 (20)  | 268 (20)  | 278 (20)  | 284 (20)  | 436*(20)  | 447 (20)  |  |
|                     | 450 (20)  | 450 (20)  | 460*(20)  | 470 (20)  | 471 (20)  | 479*(20)  | 502 (20)  | 502 (20)  |  |
|                     | 509 (20)  | 509 (20)  | 555*(20)  | 560=(20)  | 564 (20)  | 571 (20)  | 583 (20)  | 583 (20)  |  |
|                     | 584 (20)  | 585 (20)  | 651*(20)  | 1029*(21) | 1032 (21) | 1035 (21) | 1036=(21) | 1036 (21) |  |
|                     | 1038=(21) | 1038 (21) | 1040 (21) | 297*(23)  | 181*(39)  | 198 (39)  | 207 (39)  | 208 (39)  |  |
|                     | 208 (39)  | 210 (39)  | 210 (39)  | 215 (39)  | 215 (39)  | 217 (39)  | 217 (39)  |           |  |
| deltaCols           | 1168*(10) | 1188=(10) | 1190 (10) | 1196 (10) |           |           |           |           |  |
| deltaCount          | 705*(16)  | 806=(16)  | 811=(16)  | 822 (16)  |           |           |           |           |  |
| deltaH              | 1839*(13) | 2519*(13) | 555*(16)  | 574=(19)  | 603=(19)  | 605=(19)  | 613 (19)  | 457*(20)  |  |
|                     | 463 (20)  | 468 (20)  | 470 (20)  | 34*(22)   | 36 (22)   |           |           |           |  |
| deltaRows           | 1252*(10) | 1272=(10) | 1274 (10) | 1280 (10) |           |           |           |           |  |
| deltaSz             | 228*(21)  | 234 (21)  | 235 (21)  | 237 (21)  | 239 (21)  |           |           |           |  |
| deltaV              | 1839*(13) | 2519*(13) | 556*(19)  | 575=(19)  | 598=(19)  | 600=(19)  | 613 (19)  | 457*(20)  |  |
|                     | 463 (20)  | 466 (20)  | 470 (20)  | 34*(22)   | 36 (22)   |           |           |           |  |
| DeltaValue          | 2246*(13) | 466 (20)  | 468 (20)  | 1029*(21) | 1071 (21) | 1422 (21) |           |           |  |
| descent             | 1265 (6)  | 2360 (10) | 326 (40)  | 451 (40)  | 1417 (40) |           |           |           |  |
| desiredEnd          | 102*(18)  | 112=(18)  | 114 (18)  |           |           |           |           |           |  |
| deskAccName         | 763*(13)  | 2024*(16) | 2044 (16) | 2086 (16) | 2104 (16) | 2115 (16) |           |           |  |
| dest                | 1205*(40) | 1220 (40) | 1221 (40) | 1222 (40) | 1225 (40) | 1225 (40) | 1227 (40) | 1227 (40) |  |
| destRect            | 2622 (6)  | 2625 (6)  | 2640 (6)  | 3206 (6)  | 61 (39)   | 224 (39)  | 1272 (40) | 1564=(40) |  |
| DetachFromScroller  | 1301*(21) | 1310 (21) |           |           |           |           |           |           |  |
| DetachResource      | 2505 (6)  |           |           |           |           |           |           |           |  |
| dqPtr               | 348*(6)   | 353=(6)   | 356 (6)   |           |           |           |           |           |  |
| dn                  | 2014*(13) | 2110*(13) | 941*(19)  | 967 (19)  | 973 (19)  | 976 (19)  | 976 (19)  | 988 (19)  |  |
|                     | 378*(21)  | 383 (21)  | 52*(42)   | 54*(42)   |           |           |           |           |  |

|                       |           |           |           |           |           |           |           |           |  |
|-----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| diacritSens           | 273*(16)  | 299 (16)  |           |           |           |           |           |           |  |
| DialogPtr             | 521 (36)  | 58 (37)   |           |           |           |           |           |           |  |
| DialogSelect          | 77 (37)   |           |           |           |           |           |           |           |  |
| DialogViewTemplate    | 50 ( 5)   | 51*( 5)   | 335 ( 6)  | 354 ( 6)  |           |           |           |           |  |
| DialogViewTemplatePtr | 50*( 5)   | 317 ( 6)  | 348 ( 6)  | 353 ( 6)  |           |           |           |           |  |
| DIBadMount            | 1364 (16) |           |           |           |           |           |           |           |  |
| didChange             | 393*(36)  | 502*(37)  | 532 (37)  | 1695*(37) | 1731 (37) | 1792*(37) | 1823 (37) | 1926*(37) |  |
|                       | 1955 (37) | 1970*(37) | 1975 (37) | 2058*(37) | 2067=(37) |           |           |           |  |
| didMouseMove          | 3216*(16) | 3232 (16) |           |           |           |           |           |           |  |
| didMove               | 3112*(16) | 3311=(16) | 3313 (16) | 3322 (16) | 3325 (16) |           |           |           |  |
| didScroll             | 3114*(16) | 3310=(16) | 3313 (16) | 3316 (16) | 3325 (16) |           |           |           |  |
| diff                  | 306*(19)  | 313=(19)  | 314 (19)  | 315 (19)  | 317 (19)  | 322 (19)  |           |           |  |
| different             | 256*(19)  | 260=(19)  | 262 (19)  |           |           |           |           |           |  |
| DiffRgn               | 873 (10)  | 874 (10)  | 970 (10)  | 971 (10)  | 2006 (10) | 2012 (10) | 2029 (10) | 2912 (10) |  |
|                       | 2914 (10) | 2983 (10) | 3003 (10) | 452 (18)  |           |           |           |           |  |
| Dim                   | 2095*(13) | 274*(21)  | 328 (21)  |           |           |           |           |           |  |
| dimension             | 621*(18)  | 634=(18)  | 636=(18)  | 638=(18)  | 641=(18)  | 644=(18)  | 652=(18)  | 661 (18)  |  |
| DimState              | 2097*(13) | 2173*(13) | 288*(21)  | 659*(21)  |           |           |           |           |  |
| direction             | 1829*(13) | 350*(20)  | 354 (20)  | 357 (20)  |           |           |           |           |  |
| dirID                 | 402*(25)  | 204*(26)  | 220=(26)  | 224=(26)  |           |           |           |           |  |
| DisableItem           | 258 (30)  | 284 (30)  |           |           |           |           |           |           |  |
| diskEvt               | 651 (16)  | 2604 (16) |           |           |           |           |           |           |  |
| DiskFileChanged       | 1391*(13) | 161 (17)  | 242*(17)  | 254=(17)  | 257=(17)  |           |           |           |  |
| diskMask              | 387 (14)  |           |           |           |           |           |           |           |  |
| DismissDialog         | 90*( 5)   | 425 ( 6)  | 435*( 6)  | 463 ( 6)  | 481 ( 6)  | 579 ( 6)  |           |           |  |
| dismitter             | 90*( 5)   | 435*( 6)  | 440 ( 6)  |           |           |           |           |           |  |
| dismissing            | 84*( 5)   | 392*( 6)  | 395 ( 6)  |           |           |           |           |           |  |
| DispatchEvent         | 714*(13)  | 627*(16)  | 1416 (16) |           |           |           |           |           |  |
| displaying            | 878*(18)  | 898 (18)  | 907=(18)  | 908 (18)  | 912 (18)  |           |           |           |  |
| DispMCInfo            | 32 (16)   |           |           |           |           |           |           |           |  |
| DisposCIcon           | 1567 ( 6) |           |           |           |           |           |           |           |  |
| DisposDialog          | 397 (37)  |           |           |           |           |           |           |           |  |
| dispose               | 2847*(13) | 273*(14)  | 276 (14)  |           |           |           |           |           |  |
| DisposeControl        | 594 (21)  |           |           |           |           |           |           |           |  |
| disposeOnFree         | 1873*(13) | 1896*(13) | 1932*(13) | 16*(19)   | 36 (19)   | 135 (19)  | 193=(19)  | 229*(19)  |  |
|                       | 233=(19)  | 245 (19)  |           |           |           |           |           |           |  |
| DisposeRgn            | 457 (18)  | 459 (18)  | 927 (18)  | 1597 (18) |           |           |           |           |  |
| DisposeWindow         | 277 (14)  |           |           |           |           |           |           |           |  |
| DisposHandle          | 1291 ( 6) | 2448 ( 6) | 2823 ( 6) | 108 (16)  | 142 (16)  | 368 (16)  | 452 (16)  | 115 (17)  |  |
|                       | 168 (23)  | 197 (23)  | 160 (26)  | 125 (33)  | 240 (33)  | 339 (37)  | 1799 (37) | 2127 (37) |  |
|                       | 2129 (37) | 1094 (40) | 1157 (40) | 1476 (40) | 1507 (40) | 1508 (40) | 1530 (40) |           |  |
| DisposIfHandle        | 24 ( 2)   | 46 ( 2)   | 1315 ( 6) | 2857 ( 6) | 292 (10)  | 293 (10)  | 294 (10)  | 295 (10)  |  |
|                       | 296 (10)  | 297 (10)  | 298 (10)  | 2808 (10) | 2809 (10) | 129 (16)  | 237*(25)  | 156*(26)  |  |
|                       | 188 (40)  | 95 (41)   | 96 (41)   | 97 (41)   | 98 (41)   | 99 (41)   | 450 (41)  | 451 (41)  |  |
| DisposIfhandle        | 47 ( 2)   |           |           |           |           |           |           |           |  |
| DisposPixPat          | 1787 ( 6) |           |           |           |           |           |           |           |  |
| DisposPtr             | 239 (35)  |           |           |           |           |           |           |           |  |
| dit1                  | 1230*(16) |           |           |           |           |           |           |           |  |
| dlgHook               | 871*(13)  | 1350*(13) | 335*(16)  | 349 (16)  | 417*(16)  | 427 (16)  | 450 (16)  | 2927*(16) |  |
|                       | 2936=(16) | 911*(17)  | 918 (17)  | 928 (17)  | 1400*(17) | 1419=(17) |           |           |  |
| dlgID                 | 870*(13)  | 1348*(13) | 332*(16)  | 349 (16)  | 414*(16)  | 427 (16)  | 450 (16)  | 2926*(16) |  |
|                       | 2933=(16) | 908*(17)  | 918 (17)  | 928 (17)  | 1399*(17) | 1405=(17) |           |           |  |
| dlgLoc                | 910*(17)  | 918 (17)  | 928 (17)  |           |           |           |           |           |  |
| dlgNumber             | 1371*(37) | 1384=(37) | 1389=(37) | 1392 (37) |           |           |           |           |  |
| doAll                 | 2589 ( 6) | 318 (40)  | 442 (40)  | 1384 (40) |           |           |           |           |  |
| DoBreakFollowing      | 1675*(13) | 707*(18)  | 710=(18)  | 861 (37)  | 997 (37)  | 1265 (37) | 335*(38)  | 573*(40)  |  |
|                       | 631=(40)  | 633=(40)  |           |           |           |           |           |           |  |
| doc                   | 284*(16)  | 291 (16)  | 293 (16)  | 298 (16)  | 300 (16)  | 1436*(16) |           |           |  |
| DocCalcPageStrips     | 1684*(13) | 717*(18)  | 1468 (37) | 1665 (37) |           |           |           |           |  |
| DocCalcViewPerPage    | 1687*(13) | 727*(18)  | 1641 (37) | 340*(38)  | 641*(40)  | 646 (40)  |           |           |  |
| docDirID              | 288*(16)  | 294 (16)  | 297 (16)  |           |           |           |           |           |  |
| DocChangeReserve      | 303*(27)  | 325*(28)  |           |           |           |           |           |           |  |
| DocCheckPrinter       | 1691*(13) | 526 (18)  | 545 (18)  | 737*(18)  |           |           |           |           |  |
| DocChoice             | 108*( 5)  | 186*( 5)  | 226*( 5)  | 273*( 5)  | 447*( 6)  | 466 ( 6)  | 936*( 6)  | 940 ( 6)  |  |
|                       | 1063*( 6) | 1068 ( 6) | 1222*( 6) | 1238 ( 6) | 2280 ( 6) | 3069 ( 6) | 1556*(13) | 746*(18)  |  |
|                       | 749 (18)  | 483 (21)  | 682 (21)  |           |           |           |           |           |  |
| doClick               | 1550*(16) |           |           |           |           |           |           |           |  |
| docName               | 261*(36)  | 1009*(37) | 1014=(37) | 1016=(37) | 1017 (37) | 1021 (37) | 1024 (37) | 1372*(37) |  |
|                       | 1397 (37) | 1403 (37) | 1416 (37) |           |           |           |           |           |  |
| doColor               | 1397 (40) | 534 (41)  |           |           |           |           |           |           |  |
| DoCommandKey          | 92*( 5)   | 474*( 6)  | 483=( 6)  | 486=( 6)  | 486 ( 6)  | 637*(13)  | 943*(13)  | 201*(15)  |  |
|                       | 204=(15)  | 204 (15)  | 206=(15)  | 670*(16)  | 672=(16)  | 676=(16)  | 1537 (16) |           |  |
| DoCreateViews         | 650*(13)  | 1027 (14) | 105*(15)  | 124=(15)  | 124 (15)  | 176 (15)  | 192=(15)  |           |  |
| docToDelete           | 899*(13)  | 607*(16)  | 609 (16)  | 610 (16)  |           |           |           |           |  |
| documentProc          | 1270 (16) |           |           |           |           |           |           |           |  |
| docVRefnum            | 287*(16)  | 293=(16)  | 294 (16)  | 296 (16)  |           |           |           |           |  |
| DoDrawPageBreak       | 1698*(13) | 766*(18)  | 735 (37)  |           |           |           |           |           |  |
| DoDrawPrintFeedback   | 1695*(13) | 756*(18)  | 916 (18)  |           |           |           |           |           |  |
| doFace                | 1395 (40) |           |           |           |           |           |           |           |  |
| DoFailure             | 21*( 8)   | 174 ( 8)  |           |           |           |           |           |           |  |
| doFirstClick          | 1871*(13) | 1900*(13) | 133 (19)  | 191=(19)  |           |           |           |           |  |
| DoFocus               | 3161*(16) | 3273 (16) | 3329 (16) |           |           |           |           |           |  |
| doFont                | 1389 (40) |           |           |           |           |           |           |           |  |
| DoHandleEvent         | 615*(13)  | 213*(15)  | 216=(15)  | 1339 (16) |           |           |           |           |  |
| DoHighlightSelection  | 220*( 9)  | 542*(10)  | 1594*(13) | 320 (18)  | 325 (18)  | 776*(18)  | 914 (18)  |           |  |
| DoHilite              | 248*( 9)  | 553 (10)  | 563*(10)  | 2007 (10) | 2013 (10) | 2018 (10) | 2913 (10) | 2917 (10) |  |
|                       | 2978 (10) | 2984 (10) | 3004 (10) | 3008 (10) |           |           |           |           |  |
| DoIdle                | 582*( 5)  | 3007*( 6) | 3011 ( 6) | 613*(13)  | 224*(15)  | 1765 (16) | 211*(38)  | 675*(40)  |  |
| DoIdleAction          | 1754*(16) | 1794 (16) | 1795 (16) |           |           |           |           |           |  |
| DoInitialState        | 1165*(13) | 2151 (16) | 265*(17)  | 1001 (17) |           |           |           |           |  |
| DoInitUMemory         | 393*(38)  | 730 (28)  |           |           |           |           |           |           |  |
| DoInMacPrint          | 265*(26)  | 624*(37)  | 1295 (37) | 1335 (37) | 1348 (37) | 1549 (37) | 1723 (37) | 2081 (37) |  |
| DoIt                  | 2474*(13) | 53*(22)   | 498*(36)  | 2138*(37) | 408*(38)  | 474*(38)  | 523*(38)  | 574*(38)  |  |
|                       | 263*(41)  | 345*(41)  | 567*(41)  | 602 (41)  | 880*(41)  |           |           |           |  |
| DoIt                  | 2292 (16) |           |           |           |           |           |           |           |  |
| DoKeyCommand          | 95*( 5)   | 584*( 5)  | 493*( 6)  | 498=( 6)  | 500=( 6)  | 502=( 6)  | 502 ( 6)  | 3018*( 6) |  |
|                       | 3026 ( 6) | 3029=( 6) | 3032 ( 6) | 3032 ( 6) | 633*(13)  | 233*(15)  | 236=(15)  | 236 (15)  |  |
|                       | 238=(15)  | 1539 (16) | 178*(38)  | 695*(40)  | 701=(40)  | 712 (40)  | 720=(40)  | 721 (40)  |  |
|                       | 726 (40)  | 733 (40)  | 744=(40)  |           |           |           |           |           |  |



|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| DoMainFunction      | 433*(38)  | 249*(41)  | 267 (41)  | 304 (41)  | 400 (41)  |           |           |           |  |
| DoMakeDocument      | 838*(13)  | 684*(16)  | 2150 (16) | 2219 (16) | 2449 (16) |           |           |           |  |
| DoMakeEditCommand   | 182*(38)  | 720 (40)  | 757*(40)  | 770-(40)  | 777-(40)  | 784-(40)  | 842 (40)  | 857 (40)  |  |
| DoMakeStyleCommand  | 185*(38)  | 795*(40)  | 804-(40)  |           |           |           |           |           |  |
| DoMakeTypingCommand | 189*(38)  | 742 (40)  | 812*(40)  | 820-(40)  |           |           |           |           |  |
| DoMakeViews         | 1174*(13) | 2152 (16) | 2222 (16) | 2451 (16) | 275*(17)  |           |           |           |  |
| DoMakeWindows       | 1170*(13) | 2153 (16) | 2223 (16) | 290*(17)  |           |           |           |           |  |
| DoMenuCommand       | 586*( 5)  | 3039*( 6) | 3047 ( 6) | 3050-( 6) | 3053-( 6) | 3053 ( 6) | 642*(13)  | 994*(13)  |  |
|                     | 1404*(13) | 1702*(13) | 245*(15)  | 248-(15)  | 248 (15)  | 254-(15)  | 701*(16)  | 719-(16)  |  |
|                     | 831-(16)  | 831 (16)  | 2010 (16) | 299*(17)  | 314-(17)  | 320-(17)  | 320 (17)  | 343-(17)  |  |
|                     | 343 (17)  | 784*(18)  | 790-(18)  | 790 (18)  | 792-(18)  | 792 (18)  | 794-(18)  | 794 (18)  |  |
|                     | 401*(36)  | 654*(37)  | 659-(37)  | 666-(37)  | 672-(37)  | 675-(37)  | 684-(37)  | 684 (37)  |  |
|                     | 192*(38)  | 828*(40)  | 837-(40)  | 842-(40)  | 857-(40)  | 873-(40)  | 873 (40)  |           |  |
| DoHouseCommand      | 469*( 5)  | 588*( 5)  | 2239*( 6) | 2254-( 6) | 3061*( 6) | 3074 ( 6) | 3077-( 6) | 3080-( 6) |  |
|                     | 223*( 9)  | 619*(10)  | 644-(10)  | 653-(10)  | 662-(10)  | 671-(10)  | 675-(10)  | 1625*(13) |  |
|                     | 2099*(13) | 2175*(13) | 2248*(13) | 2304*(13) | 802*(18)  | 805-(18)  | 1333 (18) | 302*(21)  |  |
|                     | 310-(21)  | 676*(21)  | 683-(21)  | 1048*(21) | 1074-(21) | 1341*(21) | 1374-(21) | 1377-(21) |  |
|                     | 1377 (21) | 195*(38)  | 883*(40)  | 893-(40)  |           |           |           |           |  |
| DoMultiClick        | 622*(13)  | 262*(15)  | 265-(15)  | 265 (15)  | 267-(15)  | 592 (16)  |           |           |  |
| done                | 846*(37)  | 856-(37)  | 857 (37)  | 860-(37)  | 865-(37)  |           |           |           |  |
| DoNeedDiskSpace     | 1238*(13) | 352*(17)  | 1128 (17) |           |           |           |           |           |  |
| DoneTyping          | 199*(38)  | 227 (40)  | 708 (40)  | 866 (40)  | 889 (40)  | 901*(40)  | 1193 (40) |           |  |
| DoneViewRsrc        | 2931*(13) | 64*(18)   |           |           |           |           |           |           |  |
| DoneWithTempRgn     | 2826*(13) | 124*(14)  | 3061 (16) | 167 (18)  | 593 (18)  | 1595 (18) | 493 (19)  | 610 (19)  |  |
|                     | 513 (20)  | 1005 (40) | 1021 (40) |           |           |           |           |           |  |
| DoNormalChar        | 562*(38)  | 677*(41)  | 843 (41)  |           |           |           |           |           |  |
| dontCare            | 55*( 6)   | 60 ( 6)   | 573*( 6)  | 89*(14)   | 112-(14)  | 1665*(16) | 1749*(16) | 1800-(16) |  |
|                     | 1451*(37) | 414*(40)  | 443-(40)  |           |           |           |           |           |  |
| dontDispose         | 318*(37)  | 321-(37)  | 322 (37)  | 327-(37)  | 328 (37)  | 332-(37)  | 334 (37)  | 335-(37)  |  |
|                     | 337 (37)  |           |           |           |           |           |           |           |  |
| DoPagination        | 1705*(13) | 386 (18)  | 812*(18)  | 1730 (18) | 1561 (37) |           |           |           |  |
| DoPrint             | 1434*(37) | 1549 (37) |           |           |           |           |           |           |  |
| DoPrinterChanged    | 1709*(13) | 822*(18)  | 512 (37)  | 579 (37)  |           |           |           |           |  |
| DoRead              | 1179*(13) | 365*(17)  | 869 (17)  |           |           |           |           |           |  |
| DoScroll            | 1833*(13) | 254*(20)  | 471 (20)  | 543 (20)  | 584 (20)  |           |           |           |  |
| doScrolling         | 267*(38)  | 201*(40)  | 207 (40)  |           |           |           |           |           |  |
| DoSelectEditText    | 98*( 5)   | 457 ( 6)  | 509*( 6)  | 603 ( 6)  | 706 ( 6)  |           |           |           |  |
| DoSetCursor         | 1632*(13) | 841*(18)  | 844-(18)  | 1287 (18) | 214*(38)  | 911*(40)  | 923-(40)  | 926-(40)  |  |
| DoSetMax            | 803*(21)  | 809 (21)  |           |           |           |           |           |           |  |
| DoSetMin            | 821*(21)  | 827 (21)  |           |           |           |           |           |           |  |
| DoSetPageOffset     | 1713*(13) | 832*(18)  | 1860 (37) | 343*(38)  | 934*(40)  | 939 (40)  |           |           |  |
| DoSetText           | 839*(21)  | 845 (21)  |           |           |           |           |           |           |  |
| DoSetup             | 2844*(16) | 2919 (16) |           |           |           |           |           |           |  |
| DoSetUpMenus        | 402 (17)  |           |           |           |           |           |           |           |  |
| DoSetupMenus        | 591*( 5)  | 3088*( 6) | 3092 ( 6) | 3094 ( 6) | 646*(13)  | 997*(13)  | 1405*(13) | 1717*(13) |  |
|                     | 1996*(13) | 276*(15)  | 279 (15)  | 839*(16)  | 844 (16)  | 2854 (16) | 388*(17)  | 390 (17)  |  |
|                     | 851*(18)  | 853 (18)  | 856 (18)  | 432*(19)  | 436 (19)  | 404*(36)  | 693*(37)  | 695 (37)  |  |
|                     | 202*(38)  | 950*(40)  | 955 (40)  |           |           |           |           |           |  |
| DoSetVal            | 857*(21)  | 863 (21)  |           |           |           |           |           |           |  |
| DoSetValues         | 875*(21)  | 883 (21)  |           |           |           |           |           |           |  |
| doSize              | 1401 (40) |           |           |           |           |           |           |           |  |
| DoSubstitution      | 530*( 5)  | 593*( 5)  | 2779*( 6) | 2796-( 6) | 2813 ( 6) | 3102*( 6) | 3106-( 6) |           |  |
| DoToBreak           | 232*(36)  | 840*(37)  | 860 (37)  | 865 (37)  |           |           |           |           |  |
| DoToCell            | 262*( 9)  | 295*( 9)  | 299*( 9)  | 303*( 9)  | 309*(10)  | 320 (10)  | 1082*(10) | 1114 (10) |  |
|                     | 1125*(10) | 1127 (10) | 1139*(10) | 1147 (10) | 2574*(10) | 2580 (10) |           |           |  |
| DoToControl         | 2202*(13) | 890*(21)  | 900 (21)  | 906 (21)  | 911 (21)  |           |           |           |  |
| DoToDoc             | 907*(13)  | 1051*(16) | 1053 (16) |           |           |           |           |           |  |
| DoToEditText        | 100*( 5)  | 538*( 6)  | 547 ( 6)  |           |           |           |           |           |  |
| DoToEntry           | 71*( 1)   | 129*( 2)  | 132 ( 2)  |           |           |           |           |           |  |
| DoToEvtHandler      | 800*(13)  | 882*(16)  | 896 (16)  |           |           |           |           |           |  |
| DoToField           | 38*( 1)   | 86*( 1)   | 55*( 2)   | 59 ( 2)   | 60 ( 2)   | 61 ( 2)   | 62 ( 2)   | 355*( 2)  |  |
|                     | 360 ( 2)  | 361 ( 2)  | 362 ( 2)  | 123*( 5)  | 284*( 5)  | 330*( 5)  | 376*( 5)  | 421*( 5)  |  |
|                     | 490*( 5)  | 546*( 5)  | 623*( 5)  | 665*( 5)  | 688*( 5)  | 754*( 6)  | 758 ( 6)  | 759 ( 6)  |  |
|                     | 760 ( 6)  | 761 ( 6)  | 762 ( 6)  | 763 ( 6)  | 764 ( 6)  | 765 ( 6)  | 766 ( 6)  | 1367*( 6) |  |
|                     | 1371 ( 6) | 1372 ( 6) | 1373 ( 6) | 1374 ( 6) | 1375 ( 6) | 1590*( 6) | 1594 ( 6) | 1595 ( 6) |  |
|                     | 1596 ( 6) | 1597 ( 6) | 1598 ( 6) | 1810*( 6) | 1814 ( 6) | 1815 ( 6) | 1816 ( 6) | 1817 ( 6) |  |
|                     | 1818 ( 6) | 2002*( 6) | 2006 ( 6) | 2007 ( 6) | 2008 ( 6) | 2009 ( 6) | 2527*( 6) | 2531 ( 6) |  |
|                     | 2532 ( 6) | 2533 ( 6) | 2534 ( 6) | 2535 ( 6) | 2536 ( 6) | 2537 ( 6) | 2561*( 6) | 2565 ( 6) |  |
|                     | 2566 ( 6) | 2567 ( 6) | 2902*( 6) | 2906 ( 6) | 2907 ( 6) | 2908 ( 6) | 2909 ( 6) | 2910 ( 6) |  |
|                     | 2911 ( 6) | 3134*( 6) | 3138 ( 6) | 3139 ( 6) | 3140 ( 6) | 3141 ( 6) | 3142 ( 6) | 3540*( 6) |  |
|                     | 3544 ( 6) | 3545 ( 6) | 3546 ( 6) | 3547 ( 6) | 397*( 9)  | 494*( 9)  | 622*( 9)  | 659*( 9)  |  |
|                     | 697*( 9)  | 720*( 9)  | 743*( 9)  | 766*( 9)  | 2138*(10) | 2142 (10) | 2143 (10) | 2144 (10) |  |
|                     | 2145 (10) | 2146 (10) | 2147 (10) | 2148 (10) | 2149 (10) | 2150 (10) | 2151 (10) | 2152 (10) |  |
|                     | 2153 (10) | 2154 (10) | 2155 (10) | 2156 (10) | 2157 (10) | 2158 (10) | 2159 (10) | 2160 (10) |  |
|                     | 2161 (10) | 2162 (10) | 2163 (10) | 2164 (10) | 2165 (10) | 2166 (10) | 2167 (10) | 2168 (10) |  |
|                     | 2169 (10) | 2170 (10) | 2171 (10) | 2173 (10) | 2405*(10) | 2408 (10) | 2409 (10) | 2410 (10) |  |
|                     | 2412 (10) | 2415 (10) | 2761*(10) | 2764 (10) | 2765 (10) | 2832*(10) | 2835 (10) | 2836 (10) |  |
|                     | 2837 (10) | 2838 (10) | 2840 (10) | 3058*(10) | 3061 (10) | 3062 (10) | 3063 (10) | 3064 (10) |  |
|                     | 3066 (10) | 3093*(10) | 3096 (10) | 3097 (10) | 3099 (10) | 3126*(10) | 3129 (10) | 3130 (10) |  |
|                     | 3132 (10) | 3161*(10) | 3164 (10) | 3165 (10) | 3166 (10) | 3168 (10) | 247*(11)  | 620*(12)  |  |
|                     | 628 (12)  | 629 (12)  | 630 (12)  | 631 (12)  | 632 (12)  | 633 (12)  | 634 (12)  | 639 (12)  |  |
|                     | 641 (12)  | 607*(13)  | 697*(13)  | 1159*(13) | 1731*(13) | 1855*(13) | 2031*(13) | 2138*(13) |  |
|                     | 2205*(13) | 2270*(13) | 2314*(13) | 2346*(13) | 2466*(13) | 2580*(13) | 39*(15)   | 43 (15)   |  |
|                     | 44 (15)   | 45 (15)   | 46 (15)   | 47 (15)   | 906*(16)  | 911 (16)  | 912 (16)  | 913 (16)  |  |
|                     | 914 (16)  | 915 (16)  | 916 (16)  | 917 (16)  | 918 (16)  | 919 (16)  | 920 (16)  | 921 (16)  |  |
|                     | 922 (16)  | 923 (16)  | 924 (16)  | 925 (16)  | 926 (16)  | 927 (16)  | 928 (16)  | 929 (16)  |  |
|                     | 930 (16)  | 931 (16)  | 932 (16)  | 933 (16)  | 934 (16)  | 935 (16)  | 936 (16)  | 937 (16)  |  |
|                     | 938 (16)  | 939 (16)  | 940 (16)  | 941 (16)  | 942 (16)  | 943 (16)  | 944 (16)  | 945 (16)  |  |
|                     | 946 (16)  | 947 (16)  | 948 (16)  | 949 (16)  | 950 (16)  | 951 (16)  | 952 (16)  | 953 (16)  |  |
|                     | 954 (16)  | 955 (16)  | 956 (16)  | 957 (16)  | 958 (16)  | 959 (16)  | 960 (16)  | 961 (16)  |  |
|                     | 962 (16)  | 963 (16)  | 964 (16)  | 965 (16)  | 966 (16)  | 967 (16)  | 968 (16)  | 969 (16)  |  |
|                     | 970 (16)  | 972 (16)  | 974 (16)  | 975 (16)  | 976 (16)  | 977 (16)  | 978 (16)  | 979 (16)  |  |
|                     | 980 (16)  | 981 (16)  | 982 (16)  | 984 (16)  | 986 (16)  | 987 (16)  | 988 (16)  | 989 (16)  |  |
|                     | 990 (16)  | 991 (16)  | 992 (16)  | 993 (16)  | 994 (16)  | 995 (16)  | 996 (16)  | 997 (16)  |  |
|                     | 998 (16)  | 1000 (16) | 1002 (16) | 1003 (16) | 1004 (16) | 1005 (16) | 1006 (16) | 1007 (16) |  |
|                     | 1008 (16) | 1009 (16) | 1010 (16) | 1011 (16) | 1013 (16) | 1484*(17) | 1489 (17) | 1490 (17) |  |
|                     | 1491 (17) | 1492 (17) | 1493 (17) | 1494 (17) | 1495 (17) | 1496 (17) | 1497 (17) | 1498 (17) |  |
|                     | 1499 (17) | 1500 (17) | 1501 (17) | 1502 (17) | 1503 (17) | 1504 (17) | 1505 (17) | 1506 (17) |  |
|                     | 1507 (17) | 1508 (17) | 1509 (17) | 1510 (17) | 1511 (17) | 1512 (17) | 1513 (17) | 1514 (17) |  |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
|                   | 1902*(18) | 1906 (18) | 1907 (18) | 1908 (18) | 1909 (18) | 1910 (18) | 1911 (18) | 1912 (18) |  |
|                   | 1913 (18) | 1914 (18) | 1915 (18) | 1916 (18) | 1917 (18) | 1918 (18) | 1919 (18) | 1023*(19) |  |
|                   | 1030 (19) | 1031 (19) | 1032 (19) | 1033 (19) | 1034 (19) | 1035 (19) | 1036 (19) | 1037 (19) |  |
|                   | 1038 (19) | 1039 (19) | 1040 (19) | 1041 (19) | 1042 (19) | 1043 (19) | 1044 (19) | 1045 (19) |  |
|                   | 1046 (19) | 1047 (19) | 1048 (19) | 1049 (19) | 1050 (19) | 1051 (19) | 1052 (19) | 1053 (19) |  |
|                   | 294*(20)  | 299 (20)  | 300 (20)  | 301 (20)  | 302 (20)  | 303 (20)  | 304 (20)  | 305 (20)  |  |
|                   | 306 (20)  | 307 (20)  | 308 (20)  | 309 (20)  | 502*(21)  | 506 (21)  | 507 (21)  | 508 (21)  |  |
|                   | 509 (21)  | 510 (21)  | 511 (21)  | 512 (21)  | 513 (21)  | 514 (21)  | 516 (21)  | 518 (21)  |  |
|                   | 922*(21)  | 926 (21)  | 927 (21)  | 928 (21)  | 1180*(21) | 1185 (21) | 1186 (21) | 1187 (21) |  |
|                   | 1188 (21) | 1189 (21) | 1190 (21) | 1191 (21) | 1430*(21) | 1435 (21) | 1436 (21) | 1437 (21) |  |
|                   | 119*(22)  | 124 (22)  | 125 (22)  | 126 (22)  | 127 (22)  | 128 (22)  | 129 (22)  | 130 (22)  |  |
|                   | 131 (22)  | 132 (22)  | 133 (22)  | 134 (22)  | 135 (22)  | 136 (22)  | 134*(23)  | 139 (23)  |  |
|                   | 140 (23)  | 141 (23)  | 142 (23)  | 143 (23)  | 144 (23)  | 38*(24)   | 42 (24)   | 43 (24)   |  |
|                   | 44 (24)   | 45 (24)   | 46 (24)   | 47 (24)   | 356*(25)  | 1095*(26) | 1099 (26) | 1100 (26) |  |
|                   | 1101 (26) | 1102 (26) | 1103 (26) | 96*(31)   | 289*(32)  | 293 (32)  | 466*(36)  | 503*(36)  |  |
|                   | 874*(37)  | 878 (37)  | 879 (37)  | 880 (37)  | 881 (37)  | 882 (37)  | 883 (37)  | 884 (37)  |  |
|                   | 885 (37)  | 886 (37)  | 887 (37)  | 888 (37)  | 889 (37)  | 890 (37)  | 891 (37)  | 892 (37)  |  |
|                   | 893 (37)  | 894 (37)  | 895 (37)  | 896 (37)  | 897 (37)  | 898 (37)  | 899 (37)  | 900 (37)  |  |
|                   | 901 (37)  | 902 (37)  | 903 (37)  | 2149*(37) | 2153 (37) | 2154 (37) | 2155 (37) | 2156 (37) |  |
|                   | 356*(38)  | 448*(38)  | 532*(38)  | 586*(38)  | 1633*(40) | 1638 (40) | 1639 (40) | 1640 (40) |  |
|                   | 1641 (40) | 1642 (40) | 1643 (40) | 1644 (40) | 1645 (40) | 1646 (40) | 1648 (40) | 1650 (40) |  |
|                   | 1651 (40) | 1652 (40) | 1653 (40) | 1654 (40) | 1655 (40) | 1656 (40) | 1657 (40) | 1658 (40) |  |
|                   | 1659 (40) | 161*(41)  | 165 (41)  | 166 (41)  | 167 (41)  | 168 (41)  | 169 (41)  | 170 (41)  |  |
|                   | 171 (41)  | 172 (41)  | 173 (41)  | 174 (41)  | 175 (41)  | 176 (41)  | 177 (41)  | 178 (41)  |  |
|                   | 179 (41)  | 610*(41)  | 614 (41)  | 615 (41)  | 617 (41)  | 618 (41)  | 620 (41)  | 937*(41)  |  |
|                   | 941 (41)  | 942 (41)  | 943 (41)  | 944 (41)  |           |           |           |           |  |
| doToggle          | 31*(38)   | 571 (41)  |           |           |           |           |           |           |  |
| DoToHandle        | 276*(27)  | 948*(28)  | 969 (28)  |           |           |           |           |           |  |
| DoToItem          | 562*( 9)  | 576*( 9)  | 579*( 9)  | 2485*(10) | 2487 (10) | 2558*(10) | 2564 (10) | 2573*(10) |  |
|                   | 2576 (10) | 115*(11)  | 237*(12)  | 283 (12)  |           |           |           |           |  |
| DoToMenu          | 210*(29)  | 166*(30)  | 209 (30)  | 214 (30)  | 230 (30)  |           |           |           |  |
| DoToSubView       | 937*(18)  | 940 (18)  |           |           |           |           |           |           |  |
| DoToSubView       | 1520*(13) |           |           |           |           |           |           |           |  |
| DoToView          | 1421*(13) | 436*(17)  | 443 (17)  | 446 (17)  |           |           |           |           |  |
| DoToWind          | 910*(13)  | 1426*(13) | 1060*(16) | 1066 (16) | 1070 (16) | 454*(17)  | 456 (17)  |           |  |
| DoToWindow        | 904*(13)  | 872*(16)  | 874 (16)  |           |           |           |           |           |  |
| DoToYourWindows   | 1064*(16) | 1069 (16) |           |           |           |           |           |           |  |
| DoWrite           | 1181*(13) | 410*(17)  | 707 (17)  |           |           |           |           |           |  |
| dParams           | 379*(14)  | 439*(14)  | 440 (14)  | 441 (14)  | 480*(14)  | 514*(14)  | 515 (14)  | 517 (14)  |  |
|                   | 518 (14)  | 520 (14)  |           |           |           |           |           |           |  |
| DragWindow        | 791 (19)  |           |           |           |           |           |           |           |  |
| Draw              | 271*( 5)  | 323*( 5)  | 369*( 5)  | 414*( 5)  | 472*( 5)  | 532*( 5)  | 595*( 5)  | 1245*( 6) |  |
|                   | 1294 ( 6) | 1531*( 6) | 1556 ( 6) | 1750*( 6) | 1776 ( 6) | 1950*( 6) | 1972 ( 6) | 2291*( 6) |  |
|                   | 2328 ( 6) | 2803*( 6) | 2826 ( 6) | 2892 ( 6) | 3113*( 6) | 3121 ( 6) | 3126 ( 6) | 3299 ( 6) |  |
|                   | 227*( 9)  | 686*(10)  | 1592*(13) | 1981*(13) | 2102*(13) | 2178*(13) | 2307*(13) | 2572*(13) |  |
|                   | 863*(18)  | 910 (18)  | 1856 (18) | 444*(19)  | 317*(21)  | 690*(21)  | 1328 (21) | 1384*(21) |  |
|                   | 1397 (21) | 229*(23)  | 218*(38)  | 982*(40)  |           |           |           |           |  |
| DrawABreak        | 725*(37)  | 729*(37)  | 732*(37)  | 759 (37)  |           |           |           |           |  |
| DrawCell          | 255*( 9)  | 479*( 9)  | 807 (10)  | 823*(10)  | 2315*(10) |           |           |           |  |
| DrawCmd           | 420*(18)  | 447 (18)  | 461 (18)  |           |           |           |           |           |  |
| DrawContents      | 677*( 5)  | 2550*( 6) | 1590*(13) | 1979*(13) | 871*(18)  | 899 (18)  | 1756 (18) | 455*(19)  |  |
|                   | 458 (19)  | 519 (20)  | 294 (21)  | 397 (21)  | 408 (21)  | 831 (37)  |           |           |  |
| DrawDialog        | 1417 (37) | 2002 (37) |           |           |           |           |           |           |  |
| DrawGrowIcon      | 489 (19)  |           |           |           |           |           |           |           |  |
| DrawLabel         | 474*( 5)  | 2265 ( 6) | 2275 ( 6) | 2320 ( 6) | 2335*( 6) |           |           |           |  |
| DrawLine          | 467*(18)  | 499 (18)  | 500 (18)  | 507 (18)  | 509 (18)  | 511 (18)  | 513 (18)  |           |  |
| DrawMenuBar       | 504 (30)  |           |           |           |           |           |           |           |  |
| DrawPageBreak     | 2404*(13) | 769 (18)  | 100*(24)  | 427*(36)  | 769*(37)  |           |           |           |  |
| DrawPageInterior  | 306*(36)  | 829*(37)  | 1598 (37) |           |           |           |           |           |  |
| DrawPicture       | 1968 ( 6) | 263 (23)  |           |           |           |           |           |           |  |
| DrawPopupBox      | 476*( 5)  | 2315 ( 6) | 2358*( 6) |           |           |           |           |           |  |
| DrawPrintFeedback | 2401*(13) | 759 (18)  | 90*(24)   | 424*(36)  | 712*(37)  |           |           |           |  |
| DrawRangeOfCells  | 252*( 9)  | 719 (10)  | 767*(10)  |           |           |           |           |           |  |
| DrawResizeIcon    | 1983*(13) | 288 (19)  | 461 (19)  | 469*(19)  |           |           |           |           |  |
| drawRgn           | 424*(18)  | 443*(18)  | 448 (18)  | 450 (18)  | 452 (18)  | 456 (18)  | 457 (18)  |           |  |
| drawScrollBar     | 1819*(13) | 602*(20)  | 619 (20)  |           |           |           |           |           |  |
| drawString        | 2403 ( 6) | 2324 (10) | 373 (37)  | 807 (37)  | 811 (37)  | 815 (37)  | 819 (37)  |           |  |
| DrawSubView       | 883*(18)  | 926 (18)  |           |           |           |           |           |           |  |
| DrawWithShadow    | 420*(18)  | 488 (18)  | 491 (18)  |           |           |           |           |           |  |
| driverEvt         | 2612 (16) |           |           |           |           |           |           |           |  |
| driverHandle      | 156*(37)  | 158*(37)  | 162 (37)  | 162 (37)  | 163 (37)  |           |           |           |  |
| driverName        | 154*(37)  | 163*(37)  | 165*(37)  | 504*(37)  | 526 (37)  | 527 (37)  | 531 (37)  | 531 (37)  |  |
| drvH              | 2028*(16) | 2086*(16) | 2089 (16) | 2097 (16) | 2099 (16) | 2118 (16) |           |           |  |
| dskFullErr        | 1175 (17) |           |           |           |           |           |           |           |  |
| dstP              | 553*(12)  | 560*(12)  | 569*(12)  | 570*(12)  | 570 (12)  |           |           |           |  |
| dstRect           | 58*(42)   | 68*(42)   | 70*(42)   |           |           |           |           |           |  |
| dstVPt            | 35*(42)   | 37*(42)   |           |           |           |           |           |           |  |
| dummy             | 842*(10)  | 892 (10)  | 903 (10)  | 939*(10)  | 989 (10)  | 1000 (10) |           |           |  |
| dummyEvent        | 1124*(16) | 1169 (16) |           |           |           |           |           |           |  |
| dummyView         | 950*(18)  | 961*(18)  | 971*(18)  | 1266*(18) | 1284*(18) |           |           |           |  |
| DumpTERecord      | 613*(38)  | 52*(39)   | 405 (41)  |           |           |           |           |           |  |
| DumpTECommand     | 615*(38)  | 100*(39)  | 270 (41)  | 290 (41)  | 307 (41)  | 406 (41)  | 574 (41)  | 591 (41)  |  |
|                   | 869 (41)  | 885 (41)  | 900 (41)  | 927 (41)  |           |           |           |           |  |
| duration          | 239*( 8)  | 243*( 8)  |           |           |           |           |           |           |  |
| dv                | 2014*(13) | 2110*(13) | 941*(19)  | 969 (19)  | 973 (19)  | 977 (19)  | 977 (19)  | 989 (19)  |  |
|                   | 378*(21)  | 383 (21)  | 52*(42)   | 54*(42)   |           |           |           |           |  |
| -E-               |           |           |           |           |           |           |           |           |  |
| e                 | 172*( 4)  | 183*( 4)  | 189*( 7)  | 19*( 8)   | 36*( 8)   | 41*( 8)   | 44 ( 8)   | 49*( 8)   |  |
|                   | 53 ( 8)   | 54 ( 8)   | 115*( 8)  | 120*( 8)  | 123 ( 8)  | 128*( 8)  | 132 ( 8)  | 133 ( 8)  |  |
|                   | 1659*(16) |           |           |           |           |           |           |           |  |
| Each              | 132 ( 2)  | 690 ( 6)  | 115*(11)  | 237*(12)  | 351 (12)  | 369 (12)  | 874 (16)  | 1053 (16) |  |
|                   | 446 (17)  | 456 (17)  | 1000 (17) | 940 (18)  | 1310 (21) | 1369 (21) | 1420 (21) |           |  |
| EachBreak         | 231*(36)  | 468 (37)  | 759 (37)  | 839*(37)  |           |           |           |           |  |
| EachCellDo        | 294*( 9)  | 320 (10)  | 1081*(10) | 1153 (10) |           |           |           |           |  |
| EachEditText      | 100*( 5)  | 538*( 6)  | 741 ( 6)  |           |           |           |           |           |  |
| EachEntryDo       | 71*( 1)   | 121 ( 2)  | 129*( 2)  |           |           |           |           |           |  |
| EachFreeWindow    | 904*(13)  | 872*(16)  | 1070 (16) |           |           |           |           |           |  |
| EachHandler       | 799*(13)  | 532 (16)  | 881*(16)  | 1794 (16) | 1795 (16) |           |           |           |  |

|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| EachInRgn           | 302*( 9)  | 532 (10)  | 1127 (10) | 1138*(10) | 2580 (10) |           |           |           |  |
| EachItemDo          | 575*( 9)  | 2487 (10) | 2557*(10) |           |           |           |           |           |  |
| EachMenuDo          | 210*(29)  | 166*(30)  | 498 (30)  | 500 (30)  |           |           |           |           |  |
| EachSelectedCellDo  | 299*( 9)  | 1125*(10) |           |           |           |           |           |           |  |
| EachSelectedItemDo  | 579*( 9)  | 2573*(10) |           |           |           |           |           |           |  |
| EachSubview         | 549 ( 6)  | 553 ( 6)  | 1237 ( 6) | 2764 (16) | 328 (18)  | 546 (18)  | 612 (18)  | 926 (18)  |  |
|                     | 937*(18)  | 1452 (18) | 1501 (18) | 1602 (18) | 1676 (18) |           |           |           |  |
| EachSubview         | 1520*(13) |           |           |           |           |           |           |           |  |
| EditTextTemplate    | 555 ( 5)  | 556*( 5)  | 2964 ( 6) | 2982 ( 6) |           |           |           |           |  |
| EditTextTemplatePtr | 555*( 5)  | 2957 ( 6) | 2977 ( 6) | 2982 ( 6) |           |           |           |           |  |
| edPtr               | 2977*( 6) | 2982*( 6) | 2984 ( 6) |           |           |           |           |           |  |
| elt                 | 555*(12)  | 566*(12)  | 567 (12)  | 569 (12)  |           |           |           |           |  |
| EmpowerChooser      | 541*(36)  | 137*(37)  | 195 (37)  | 2050 (37) |           |           |           |           |  |
| empty               | 34*(32)   | 63 (32)   |           |           |           |           |           |           |  |
| EmptyHandle         | 244 (28)  | 254 (28)  | 631 (28)  | 679 (28)  | 692 (28)  | 1000 (28) |           |           |  |
| EmptyRect           | 1863 (18) |           |           |           |           |           |           |           |  |
| EmptyRgn            | 507 (10)  | 550 (10)  | 1860 (10) | 3051 (16) | 273 (20)  |           |           |           |  |
| EmptyVRect          | 60*(42)   |           |           |           |           |           |           |           |  |
| Enable              | 846 (16)  | 847 (16)  | 848 (16)  | 852 (16)  | 853 (16)  | 856 (16)  | 860 (16)  | 865 (16)  |  |
|                     | 2885 (16) | 2895 (16) | 2899 (16) | 2902 (16) | 2903 (16) | 2914 (16) | 392 (17)  | 393 (17)  |  |
|                     | 396 (17)  | 397 (17)  | 215*(29)  | 243*(30)  | 699 (37)  | 700 (37)  | 701 (37)  | 964 (40)  |  |
|                     | 966 (40)  | 973 (40)  | 974 (40)  |           |           |           |           |           |  |
| EnableArray         | 420*(30)  | 425 (30)  |           |           |           |           |           |           |  |
| EnableCheck         | 2890 (16) | 2891 (16) | 2892 (16) | 2893 (16) | 2894 (16) | 2909 (16) | 2910 (16) | 218*(29)  |  |
|                     | 268*(30)  | 704 (37)  |           |           |           |           |           |           |  |
| enableFlags         | 2860 (16) | 2862*(16) | 2862 (16) | 442 (30)  | 443*(30)  | 475 (30)  | 480*(30)  |           |  |
| EnableItem          | 256 (30)  | 282 (30)  |           |           |           |           |           |           |  |
| endLoc              | 843*(37)  | 851*(37)  | 857 (37)  |           |           |           |           |           |  |
| EndMenuSetup        | 467*(30)  | 500 (30)  |           |           |           |           |           |           |  |
| endMeth             | 46*(32)   | 55*(32)   | 57 (32)   |           |           |           |           |           |  |
| endOfSelection      | 400*(40)  | 423*(40)  | 438 (40)  | 494 (40)  |           |           |           |           |  |
| endTriplet          | 228*( 8)  | 241 ( 8)  | 242 ( 8)  | 243 ( 8)  |           |           |           |           |  |
| EndUpdate           | 1757 (18) | 520 (20)  |           |           |           |           |           |           |  |
| EntDebugger         | 2830*(13) | 144*(14)  | 495 (14)  |           |           |           |           |           |  |
| entering            | 74*( 3)   | 70*( 4)   | 74 ( 4)   | 78 ( 4)   | 1542*(13) | 1957*(13) | 2087*(13) | 2302*(13) |  |
|                     | 2830*(13) | 144*(14)  | 146 (14)  | 306*(18)  | 313 (18)  | 318 (18)  | 253*(19)  | 260 (19)  |  |
|                     | 265 (19)  | 267 (19)  | 290 (19)  | 216*(21)  | 1319*(21) | 1323 (21) | 452*(36)  | 1991*(37) |  |
|                     | 1995 (37) |           |           |           |           |           |           |           |  |
| EntryWithKey        | 73*( 1)   | 139*( 2)  | 156*( 2)  | 216 ( 2)  | 1160 (14) | 73 (15)   |           |           |  |
| EntryWithValue      | 75*( 1)   | 162*( 2)  | 174*( 2)  |           |           |           |           |           |  |
| environsVersion     | 172*(25)  |           |           |           |           |           |           |           |  |
| envMac              | 115 (26)  |           |           |           |           |           |           |           |  |
| envXL               | 77 ( 4)   | 127 ( 4)  | 175 ( 4)  | 208 ( 4)  |           |           |           |           |  |
| EqualBlocks         | 240*(25)  | 166*(26)  | 740 (41)  |           |           |           |           |           |  |
| EqualPt             | 1843 (10) | 566 (37)  |           |           |           |           |           |           |  |
| EqualRect           | 564 (37)  | 565 (37)  | 1648 (37) | 1652 (37) | 1661 (37) |           |           |           |  |
| EqualString         | 298 (16)  | 527 (37)  |           |           |           |           |           |           |  |
| EqualVPT            | 3311 (16) | 1657 (37) | 1662 (37) | 41*(42)   |           |           |           |           |  |
| EqualVRect          | 62*(42)   |           |           |           |           |           |           |           |  |
| EraseRect           | 2400 ( 6) | 2891 ( 6) | 3199 ( 6) | 3298 ( 6) | 447 (19)  | 832 (19)  | 1010 (19) | 518 (20)  |  |
| err                 | 1250*( 6) | 1275*( 6) | 1276 ( 6) | 1290 ( 6) | 2782*( 6) | 2787*( 6) | 2788 ( 6) | 2796 ( 6) |  |
|                     | 2905*(13) | 155*(14)  | 208 (14)  | 213 (14)  | 1118*(14) | 1122*(14) | 1125 (14) | 1126 (14) |  |
|                     | 1128 (14) | 193*(16)  | 225*(16)  | 279*(16)  | 286*(16)  | 294*(16)  | 295 (16)  | 308*(16)  |  |
|                     | 310 (16)  | 377*(16)  | 394*(16)  | 1092*(16) | 1100*(16) | 1102 (16) | 1103 (16) | 1112 (16) |  |
|                     | 1358*(16) | 1364*(16) | 1366 (16) | 1367 (16) | 2035*(16) | 157*(17)  | 161*(17)  | 163 (17)  |  |
|                     | 172 (17)  | 173 (17)  | 244*(17)  | 248*(17)  | 249 (17)  | 251*(17)  | 253*(17)  | 254 (17)  |  |
|                     | 302*(17)  | 474*(17)  | 478*(17)  | 480 (17)  | 481 (17)  | 519*(17)  | 534*(17)  | 535 (17)  |  |
|                     | 536*(17)  | 539*(17)  | 550*(17)  | 552 (17)  | 627*(17)  | 637*(17)  | 639*(17)  | 641 (17)  |  |
|                     | 642 (17)  | 645*(17)  | 647 (17)  | 648 (17)  | 649 (17)  | 689*(17)  | 697*(17)  | 699 (17)  |  |
|                     | 811*(17)  | 813*(17)  | 815 (17)  | 816 (17)  | 914*(17)  | 944*(17)  | 945 (17)  | 945 (17)  |  |
|                     | 946 (17)  | 1056*(17) | 1066*(17) | 1144*(17) | 1146 (17) | 1168 (17) | 1169 (17) | 1181*(17) |  |
|                     | 1182 (17) | 1198*(17) | 1253*(17) | 1263*(17) | 1264 (17) | 1265 (17) | 1266 (17) | 1295*(17) |  |
|                     | 1301*(17) | 1303*(17) | 1305 (17) | 1306 (17) | 1307 (17) | 1365*(17) | 1366 (17) | 1367 (17) |  |
|                     | 1368 (17) | 1240*(18) | 1248*(18) | 1252 (18) | 1253 (18) | 1256 (18) | 1835*(18) | 1872*(18) |  |
|                     | 1875 (18) | 1880 (18) | 177*(23)  | 180*(23)  | 181 (23)  | 184 (23)  | 184 (23)  | 185 (23)  |  |
|                     | 187 (23)  | 28*(26)   | 31*(26)   | 34*(26)   | 39 (26)   | 40*(26)   | 43 (26)   | 135*(26)  |  |
|                     | 144*(26)  | 146 (26)  | 147*(26)  | 149 (26)  | 234*(26)  | 243*(26)  | 244 (26)  | 245*(26)  |  |
|                     | 246 (26)  | 590*(26)  | 593 (26)  | 595 (26)  | 903*(28)  | 161*(33)  | 202*(33)  | 208 (33)  |  |
|                     | 111*(37)  | 118*(37)  | 119 (37)  | 123 (37)  | 126 (37)  | 175*(37)  | 228*(37)  | 235 (37)  |  |
|                     | 607*(37)  | 612*(37)  | 614 (37)  | 615 (37)  | 1126*(37) | 1134*(37) | 1157 (37) | 1175 (37) |  |
|                     | 1183 (37) | 1191 (37) | 1194 (37) | 1195 (37) | 1206*(37) | 1210*(37) | 1211 (37) | 1213 (37) |  |
|                     | 1213 (37) | 1214*(37) | 1215 (37) | 1282*(37) | 1296 (37) | 1438*(37) | 1475*(37) | 1082*(40) |  |
|                     | 1118*(40) | 1120 (40) | 1121 (40) | 1150*(40) | 1154 (40) |           |           |           |  |
| errAppTable         | 240*(13)  | 771 (14)  |           |           |           |           |           |           |  |
| errFileChanged      | 284*(13)  | 163 (17)  | 253 (17)  |           |           |           |           |           |  |
| errFileTypeChanged  | 288*(13)  | 251 (17)  |           |           |           |           |           |           |  |
| errNoPrintDrvr      | 289*(13)  | 1214 (37) |           |           |           |           |           |           |  |
| errNotMyType        | 290*(13)  | 1511 (16) |           |           |           |           |           |           |  |
| errOperationsID     | 238*(13)  | 198 (14)  |           |           |           |           |           |           |  |
| error               | 21*( 2)   | 199*( 2)  | 265*( 6)  | 308*( 6)  | 1129*( 6) | 1399*( 6) | 1441*( 6) | 1622*( 6) |  |
|                     | 1664*( 6) | 1843*( 6) | 1878*( 6) | 2035*( 6) | 2089*( 6) | 2493*( 6) | 2678*( 6) | 153*( 7)  |  |
|                     | 193*( 7)  | 206*( 7)  | 218*( 7)  | 26*( 8)   | 74*( 8)   | 78 ( 8)   | 86*( 8)   | 95 ( 8)   |  |
|                     | 100*( 8)  | 104 ( 8)  | 105 ( 8)  | 141*( 8)  | 165 ( 8)  | 172*( 8)  | 172 ( 8)  | 180 ( 8)  |  |
|                     | 259*(12)  | 1065*(13) | 300*(14)  | 304 (14)  | 309 (14)  | 843*(14)  | 910*(14)  | 985*(14)  |  |
|                     | 117*(15)  | 199*(16)  | 206 (16)  | 713*(16)  | 1237*(16) | 1389*(16) | 1445*(16) | 1447 (16) |  |
|                     | 1457 (16) | 1465*(16) | 1467 (16) | 1468 (16) | 1962*(16) | 1967 (16) | 2037*(16) | 2050 (16) |  |
|                     | 2140*(16) | 2143 (16) | 2195*(16) | 2203 (16) | 2252*(16) | 2266 (16) | 2329*(16) | 2334 (16) |  |
|                     | 2338 (16) | 2440*(16) | 2495*(16) | 2501 (16) | 2946*(16) | 2948 (16) | 19*(17)   | 308*(17)  |  |
|                     | 636*(17)  | 810*(17)  | 964*(17)  | 966 (17)  | 967*(17)  | 970 (17)  | 1063*(17) | 1078 (17) |  |
|                     | 1300*(17) | 1314*(17) | 1326 (17) | 1330 (17) | 185*(18)  | 431*(18)  | 27*(19)   | 96*(19)   |  |
|                     | 214*(20)  | 631*(21)  | 1213*(21) | 1249*(21) | 165*(23)  | 139*(31)  | 10*(33)   | 253*(37)  |  |
|                     | 631*(37)  | 1131*(37) | 1134 (37) | 1458*(37) | 1531*(37) | 1535 (37) | 1535 (37) | 1539 (37) |  |
|                     | 1587*(37) | 1710*(37) | 2073*(37) | 2099*(37) | 1089*(40) | 1096 (40) | 1102 (40) | 1209*(40) |  |
|                     | 22*(41)   | 357*(41)  | 447*(41)  | 639*(41)  | 824*(41)  |           |           |           |  |
| ErrorAlert          | 2905*(13) | 155*(14)  | 309 (14)  | 2948 (16) |           |           |           |           |  |
| errReasonID         | 234*(13)  | 208 (14)  |           |           |           |           |           |           |  |
| ErrRecord           | 724 (14)  | 725*(14)  | 742 (14)  | 764 (14)  |           |           |           |           |  |
| errRecoveryID       | 236*(13)  | 215 (14)  |           |           |           |           |           |           |  |
| errRevertFNF        | 282*(13)  | 967 (17)  |           |           |           |           |           |           |  |

```

errSaveAgain      286* (13) 1211 (17)
errSpooling       281* (13) 1521 (37)
errStr            173* (14) 208 (14) 217 (14)
EventAvail        1153 (16) 1169 (16) 3336 (16)
EventInfo         92 (5) 96 (5) 104 (5) 106 (5) 469 (5) 584 (5) 588 (5) 474 (6)
                  494 (6) 570 (6) 588 (6) 2239 (6) 3018 (6) 3061 (6) 223 (9) 619 (10)
                  500* (13) 634 (13) 637 (13) 708 (13) 714 (13) 721 (13) 723 (13) 728 (13)
                  732 (13) 737 (13) 744 (13) 748 (13) 754 (13) 784 (13) 943 (13) 1621 (13)
                  1625 (13) 1988 (13) 2099 (13) 2175 (13) 2248 (13) 2304 (13) 201 (15) 233 (15)
                  627 (16) 670 (16) 1303 (16) 1328 (16) 1352 (16) 1384 (16) 1524 (16) 1547 (16)
                  1644 (16) 1656 (16) 1703 (16) 2390 (16) 2532 (16) 802 (18) 1298 (18) 685 (19)
                  302 (21) 676 (21) 1048 (21) 1341 (21) 179 (38) 196 (38) 695 (40) 883 (40)
eventMask         703* (13) 1119* (16) 1153 (16) 1160 (16) 1169 (16) 1186 (16)
EventRecord       498 (13) 704 (13) 716 (13) 1120 (16) 1124 (16) 1379 (16) 1664 (16) 2323 (16)
                  3109 (16) 3364 (16) 691 (19) 59 (37)
everyEvent        387 (14) 511 (14)
EXIT              190 (6) 330 (12) 527 (12) 746 (12) 760 (12) 263 (14) 1040 (16) 1579 (16)
                  1588 (16) 3013 (16) 3024 (16) 1014 (18) 1020 (18) 1050 (18) 596 (26) 83 (33)
                  142 (33) 191 (33) 229 (35) 1411 (37) 2076 (37) 712 (40) 721 (40) 726 (40)
                  733 (40)
Exit              77 (30) 128 (30) 119 (32) 1186 (37)
ExitMacApp        2958* (13) 233* (14)
ExitToShell       182 (8) 236 (14) 311 (14) 407 (14)
exp               3525 (6)
ExpandPtr         353 (6) 1489 (6) 1711 (6) 1915 (6) 2147 (6) 2982 (6) 3449 (6) 248 (10)
                  2935* (13) 97* (18) 128* (18) 139 (18) 88 (20) 999 (21)
ExpandPtrWStr     830 (6) 912 (6) 1039 (6) 1189 (6) 2742 (6) 2283 (10) 2942* (13) 136* (18)
                  139* (18) 244 (18) 182 (19) 174 (21) 136 (40)
extend            366* (9) 374* (9) 381* (9) 603* (9) 606* (9) 1933* (10) 1942 (10) 1985* (10)
                  2004 (10) 2022 (10) 2045* (10) 2049 (10) 2710* (10) 2712 (10) 2721* (10) 2728 (10)
                  1040 (26)
Extended          3514 (6)
extent            684* (18) 687 (18) 688 (18) 1078* (18) 1081 (18) 1082 (18) 1212* (18) 1216 (18)
                  1217 (18)
ExtractStyles     285* (38) 1037* (40)
ExtractText       3162 (6) 290* (38) 1049* (40) 1052* (40)
extValue          3514* (6) 3527* (6) 3528 (6) 3528 (6)

-F-
f0                1689* (37)
f1                1689* (37)
f10               1689* (37)
f11               1689* (37)
f12               1689* (37)
f13               1689* (37)
f14               1689* (37)
f15               1689* (37)
f2                1689* (37) 1730* (37)
f3                1689* (37)
f4                1689* (37)
f5                1689* (37)
f6                1689* (37)
f7                1689* (37)
f8                1689* (37)
f9                1689* (37)
fAcceptsChanges   124* (38) 38* (40) 102* (40) 143 (40) 680 (40) 702 (40) 961 (40) 973 (40)
                  974 (40) 1651 (40) 379* (41)
fAdapted          1922* (13) 46* (19) 196 (19) 311* (19) 1039 (19)
fAdornCols        143* (9) 54* (10) 154* (10) 266 (10) 721 (10) 2157 (10)
fAdornment        2939* (6) 3123 (6) 2067* (13) 111* (21) 141* (21) 179 (21) 266 (21) 322 (21)
                  325 (21) 512 (21) 710 (21) 713 (21)
fAdornRows        142* (9) 53* (10) 153* (10) 265 (10) 740 (10) 2156 (10)
failA6            155* (7)
FailInfo          16 (2) 194 (2) 260 (6) 303 (6) 1124 (6) 1394 (6) 1436 (6) 1617 (6)
                  1659 (6) 1838 (6) 1873 (6) 2029 (6) 2083 (6) 2487 (6) 2673 (6) 150 (7)
                  151* (7) 188 (7) 225 (7) 18 (8) 297 (8) 243 (12) 835 (14) 902 (14)
                  980 (14) 115 (15) 194 (16) 708 (16) 1232 (16) 1382 (16) 1439 (16) 1951 (16)
                  2034 (16) 2133 (16) 2190 (16) 2246 (16) 2322 (16) 2435 (16) 2490 (16) 14 (17)
                  303 (17) 631 (17) 800 (17) 959 (17) 1055 (17) 1294 (17) 1318 (17) 183 (18)
                  426 (18) 22 (19) 91 (19) 212 (20) 629 (21) 1211 (21) 1247 (21) 160 (23)
                  248 (37) 626 (37) 1127 (37) 1429 (37) 1453 (37) 1574 (37) 1697 (37) 2060 (37)
                  2094 (37) 1084 (40) 1207 (40) 17 (41) 349 (41) 442 (41) 523 (41) 634 (41)
                  819 (41)
FailMemError      221 (2) 197* (7) 34* (8) 51 (18) 123 (18) 502 (41) 750 (41) 776 (41)
                  797 (41)
FailNewMessage    206* (7) 74* (8) 1967 (16) 2050 (16) 2143 (16) 2203 (16) 2266 (16) 970 (17)
                  1078 (17) 1539 (37)
FailNIL           32 (2) 35 (2) 98 (2) 227 (2) 31 (6) 32 (6) 33 (6) 34 (6)
                  35 (6) 36 (6) 37 (6) 38 (6) 39 (6) 40 (6) 41 (6) 42 (6)
                  43 (6) 44 (6) 277 (6) 324 (6) 618 (6) 2052 (6) 2108 (6) 214* (7)
                  62* (8) 641 (10) 650 (10) 659 (10) 668 (10) 692 (14) 693 (14) 694 (14)
                  695 (14) 696 (14) 819 (14) 866 (14) 939 (14) 1004 (14) 83 (15) 130 (15)
                  347 (16) 425 (16) 1255 (16) 1286 (16) 2118 (16) 375 (17) 228 (20) 240 (20)
                  308 (21) 644 (21) 92 (23) 207 (23) 341 (30) 200 (37) 1344 (37) 1818 (37)
                  1948 (37) 2111 (37) 2115 (37) 27 (39) 768 (40) 775 (40) 782 (40) 1232 (40)
                  1609 (40) 56 (41) 64 (41) 80 (41) 372 (41) 465 (41) 469 (41) 653 (41)
FailNil           74 (10) 78 (10) 180 (10) 184 (10) 42 (12) 646 (14) 147 (16) 2508 (16)
                  324 (30) 802 (40) 818 (40)
FailNoReserve     310* (27) 527* (28) 1237 (40)
FailOSErr         143 (4) 144 (4) 146 (4) 149 (4) 154 (4) 155 (4) 218* (7) 86* (8)
                  424 (17) 665 (17) 673 (17) 674 (17) 680 (17) 699 (17) 702 (17) 711 (17)
                  848 (17) 861 (17) 891 (17) 1117 (17) 1234 (17) 1334 (17) 1370 (17) 215 (23)
                  516 (28) 1581 (37) 1443 (40) 1603 (40) 1610 (40) 149 (41) 238 (41) 679 (41)
FailOSerr         379 (17)
FailOsErr         456 (14)
failPC            156* (7)
FailResError      1143 (6) 1419 (6) 1466 (6) 1642 (6) 1688 (6) 1856 (6) 1892 (6) 2506 (6)
                  2693 (6) 203* (7) 113* (8) 2091 (16) 682 (17)
FailSpaceIsLow    2113 (16) 2117 (16) 2171 (16) 2230 (16) 2687 (16) 313* (27) 538* (28) 1168 (40)

```

[illegible]

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| fCurrentItem        | 442*( 5)  | 2046-( 6) | 2102-( 6) | 2110 ( 6) | 2156 ( 6) | 2272 ( 6) | 2313 ( 6) | 2376 ( 6) |
|                     | 2421 ( 6) | 2453-( 6) | 2464 ( 6) | 2466 ( 6) | 2467 ( 6) | 2470-( 6) | 2535 ( 6) |           |
| fDataHandle         | 251*( 5)  | 303*( 5)  | 349*( 5)  | 394*( 5)  | 510*( 5)  | 1135-( 6) | 1161-( 6) | 1258 ( 6) |
|                     | 1274 ( 6) | 1277 ( 6) | 1303 ( 6) | 1304 ( 6) | 1315 ( 6) | 1317-( 6) | 1354-( 6) | 1356-( 6) |
|                     | 1374 ( 6) | 1405-( 6) | 1413-( 6) | 1414 ( 6) | 1417-( 6) | 1447-( 6) | 1459-( 6) | 1460 ( 6) |
|                     | 1463-( 6) | 1537 ( 6) | 1542 ( 6) | 1543 ( 6) | 1547 ( 6) | 1548 ( 6) | 1549 ( 6) | 1550 ( 6) |
|                     | 1553 ( 6) | 1567 ( 6) | 1569 ( 6) | 1570-( 6) | 1580-( 6) | 1597 ( 6) | 1628-( 6) | 1636-( 6) |
|                     | 1637 ( 6) | 1640-( 6) | 1670-( 6) | 1682-( 6) | 1683 ( 6) | 1686-( 6) | 1756 ( 6) | 1762 ( 6) |
|                     | 1763 ( 6) | 1767 ( 6) | 1768 ( 6) | 1769 ( 6) | 1770 ( 6) | 1773 ( 6) | 1787 ( 6) | 1789 ( 6) |
|                     | 1790-( 6) | 1800-( 6) | 1817 ( 6) | 1849-( 6) | 1855-( 6) | 1884-( 6) | 1891-( 6) | 1956 ( 6) |
|                     | 1960 ( 6) | 1961 ( 6) | 1965 ( 6) | 1966 ( 6) | 1968 ( 6) | 1969 ( 6) | 1982-( 6) | 1992-( 6) |
|                     | 2008 ( 6) | 2684-( 6) | 2711-( 6) | 2786 ( 6) | 2789 ( 6) | 2811 ( 6) | 2836 ( 6) | 2837 ( 6) |
|                     | 2857 ( 6) | 2859-( 6) | 2882 ( 6) | 2882 ( 6) | 2885-( 6) | 2887-( 6) | 2909 ( 6) | 3105 ( 6) |
|                     | 2557*(13) | 30-(23)   | 52-(23)   | 84 (23)   | 97 (23)   | 117 (23)  | 123 (23)  | 143 (23)  |
|                     | 169-(23)  | 195 (23)  | 197 (23)  | 198-(23)  | 216-(23)  | 245 (23)  | 245 (23)  | 256 (23)  |
|                     | 263 (23)  |           |           |           |           |           |           |           |
| fDataOpen           | 1121*(13) | 49-(17)   | 53 (17)   | 476 (17)  | 839 (17)  | 858 (17)  | 875 (17)  | 1235 (17) |
|                     | 1255 (17) | 1335 (17) | 1507 (17) |           |           |           |           |           |
| fDataPerm           | 1125*(13) | 74-(17)   | 850 (17)  | 1236 (17) | 1336 (17) | 1509 (17) |           |           |
| fDataRefNum         | 860 (17)  |           |           |           |           |           |           |           |
| fDataRefNum         | 1129*(13) | 30-(17)   | 478 (17)  | 877-(17)  | 881-(17)  | 1238-(17) | 1339-(17) | 1511 (17) |
| fdCreator           | 546-(17)  | 558-(17)  | 666 (17)  |           |           |           |           |           |
| fDefaultControl     | 572*( 6)  |           |           |           |           |           |           |           |
| fDefaultItem        | 57*( 5)   | 280-( 6)  | 319-( 6)  | 358 ( 6)  | 424 ( 6)  | 425 ( 6)  | 575 ( 6)  | 576 ( 6)  |
|                     | 578 ( 6)  | 579 ( 6)  | 759 ( 6)  |           |           |           |           |           |
| fDefChoice          | 786-( 6)  | 804-( 6)  | 868-( 6)  | 886-( 6)  | 996-( 6)  | 1014-( 6) | 1148-( 6) | 1163-( 6) |
|                     | 1423-( 6) | 1469-( 6) | 1646-( 6) | 1691-( 6) | 1860-( 6) | 1895-( 6) | 2062-( 6) | 2119-( 6) |
|                     | 2280 ( 6) | 2698-( 6) | 2714-( 6) | 2941-( 6) | 2962-( 6) | 2062*(13) | 107-(21)  | 140-(21)  |
|                     | 483 (21)  | 507 (21)  | 682 (21)  | 950-(21)  | 952-(21)  |           |           |           |
| fDeletions          | 55*(11)   | 64-(12)   | 159 (12)  | 203-(12)  | 203 (12)  | 252 (12)  | 575-(12)  | 631 (12)  |
|                     | 689 (12)  | 743 (12)  |           |           |           |           |           |           |
| fDeviceRes          | 2327*(13) | 44 (24)   | 517 (37)  | 550 (37)  | 566 (37)  |           |           |           |
| fdImmed             | 2064*(13) | 109-(21)  | 144-(21)  | 182 (21)  | 290 (21)  | 292-(21)  | 327 (21)  | 417 (21)  |
|                     | 509 (21)  | 650 (21)  | 664-(21)  | 668 (21)  |           |           |           |           |
| fDirection          | 2225*(13) | 354 (20)  | 947-(21)  | 977-(21)  | 979-(21)  | 1187 (21) | 1232 (21) | 1304 (21) |
|                     | 1305 (21) | 1353 (21) | 1415 (21) |           |           |           |           |           |
| fDismissed          | 62*( 5)   | 283-( 6)  | 328-( 6)  | 439-( 6)  | 653-( 6)  | 656 ( 6)  | 764 ( 6)  |           |
| fDismisser          | 63*( 5)   | 284-( 6)  | 329-( 6)  | 423 ( 6)  | 440-( 6)  | 657 ( 6)  | 765 ( 6)  |           |
| fDismissesDialog    | 460 ( 6)  | 2066*(13) | 117-(21)  | 145-(21)  | 184 (21)  | 511 (21)  |           |           |
| fDisposeOnFree      | 1917*(13) | 36-(19)   | 106-(19)  | 135-(19)  | 193 (19)  | 233 (19)  | 1048 (19) |           |
| fDocPrintHandler    | 1082*(13) | 2456 (16) | 28-(17)   | 319 (17)  | 320 (17)  | 400 (17)  | 402 (17)  | 441 (17)  |
|                     | 442 (17)  | 443 (17)  | 790 (17)  | 1493 (17) | 301 (37)  | 302-(37)  | 330 (37)  | 331-(37)  |
| fDocument           | 1463*(13) | 195 (17)  | 193-(18)  | 294 (18)  | 295 (18)  | 1092 (18) | 1093 (18) | 1909 (18) |
|                     | 37-(19)   | 103-(19)  | 236 (19)  | 237 (19)  | 274 (19)  | 412 (19)  | 418 (19)  | 421 (19)  |
|                     | 725-(19)  | 136-(21)  | 95*(36)   | 261-(37)  | 326 (37)  | 570 (37)  | 571 (37)  | 573 (37)  |
|                     | 884 (37)  | 1013 (37) | 1014 (37) | 1805 (37) | 1806 (37) | 1806 (37) | 1808 (37) | 2108 (37) |
|                     | 50 (41)   |           |           |           |           |           |           |           |
| fDoFirstClick       | 1921*(13) | 783-(16)  | 783 (16)  | 2905 (16) | 3032 (16) | 59-(19)   | 133-(19)  | 191 (19)  |
|                     | 700 (19)  | 1052 (19) |           |           |           |           |           |           |
| fdType              | 250 (17)  | 547-(17)  | 559-(17)  | 667 (17)  |           |           |           |           |
| fEachLevel          | 59*(11)   | 63-(12)   | 201 (12)  | 223 (12)  | 250-(12)  | 250 (12)  | 251 (12)  | 267-(12)  |
|                     | 267 (12)  | 429 (12)  | 632 (12)  |           |           |           |           |           |
| fEditText           | 675*( 5)  | 623-( 6)  | 2566 ( 6) | 2606-( 6) |           |           |           |           |
| FeedbackOnce        | 3189*(16) | 3267 (16) | 3313 (16) | 3325 (16) | 3350 (16) |           |           |           |
| fEffectiveDeviceRes | 2329*(13) | 45 (24)   | 282 (37)  | 283 (37)  | 518 (37)  | 541 (37)  | 555 (37)  | 556 (37)  |
|                     | 557 (37)  | 558 (37)  |           |           |           |           |           |           |
| fElitClass          | 61*(11)   | 66-(12)   | 600 (12)  | 601 (12)  | 603 (12)  | 608-(12)  | 633 (12)  | 654 (12)  |
|                     | 655 (12)  |           |           |           |           |           |           |           |
| fEntries            | 61*( 1)   | 100-( 2)  | 102 ( 2)  | 122 ( 2)  | 132 ( 2)  | 156 ( 2)  | 184 ( 2)  | 229 ( 2)  |
|                     | 287 ( 2)  | 289 ( 2)  | 312 ( 2)  | 338 ( 2)  | 361 ( 2)  | 690 ( 6)  |           |           |
| fFileType           | 1104*(13) | 44-(17)   | 250 (17)  | 547 (17)  | 559 (17)  | 1498 (17) |           |           |
| ffFinderJobDialog   | 167*(36)  | 202-(37)  | 274-(37)  | 897 (37)  | 1941 (37) |           |           |           |
| ffFinderSetup       | 163*(36)  | 273-(37)  | 896 (37)  | 1931 (37) |           |           |           |           |
| ffFirstChar         | 548*(38)  | 656-(41)  | 882 (41)  | 943 (41)  |           |           |           |           |
| ffFirstOffset       | 53*(11)   | 62-(12)   | 157 (12)  | 228 (12)  | 271 (12)  | 320 (12)  | 456 (12)  | 517 (12)  |
|                     | 559 (12)  | 576 (12)  | 630 (12)  |           |           |           |           |           |
| ffFixedSizePages    | 103*(36)  | 268-(37)  | 269-(37)  | 459 (37)  | 886 (37)  | 887 (37)  | 979 (37)  | 1257 (37) |
| ffFocusedPage       | 2335*(13) | 361 (37)  | 1850-(37) |           |           |           |           |           |
| ffForcedOnScreen    | 1926*(13) | 50-(19)   | 198 (19)  | 560-(19)  | 1043 (19) |           |           |           |
| ffFreeOnClosing     | 1916*(13) | 35-(19)   | 105-(19)  | 134-(19)  | 192 (19)  | 387 (19)  | 1047 (19) |           |
| ffFreeText          | 130*(38)  | 43-(40)   | 105-(40)  | 144 (40)  | 187 (40)  | 1654 (40) | 395-(41)  |           |
| fgColor             | 313 (26)  |           |           |           |           |           |           |           |
| fGridView           | 640*( 9)  | 2791-(10) | 2836 (10) | 2939 (10) | 2993 (10) | 3032 (10) |           |           |
| fHavePicture        | 2552*(13) | 31-(23)   | 53-(23)   | 140 (23)  | 202-(23)  | 204 (23)  | 250 (23)  |           |
| fHaveText           | 2553*(13) | 32-(23)   | 54-(23)   | 85 (23)   | 141 (23)  | 201-(23)  | 204 (23)  | 240 (23)  |
| fHilite             | 2063*(13) | 108-(21)  | 143-(21)  | 183 (21)  | 329 (21)  | 366 (21)  | 368-(21)  | 480 (21)  |
|                     | 508 (21)  | 768 (21)  | 770-(21)  |           |           |           |           |           |
| fHlDesired          | 2001 (10) | 2007 (10) | 2013 (10) | 2018 (10) | 1467*(13) | 198-(18)  | 228-(18)  | 320 (18)  |
|                     | 321-(18)  | 325 (18)  | 326-(18)  | 914 (18)  | 1914 (18) |           |           |           |
| fHlRegion           | 151*( 9)  | 108-(10)  | 208-(10)  | 293 (10)  | 550 (10)  | 552 (10)  | 874 (10)  | 874 (10)  |
|                     | 971 (10)  | 971 (10)  | 1995 (10) | 2006 (10) | 2012 (10) | 2012 (10) | 2013 (10) | 2017 (10) |
|                     | 2017 (10) | 2018 (10) | 2032 (10) | 2144 (10) | 2884 (10) | 2908 (10) | 2920 (10) | 2963 (10) |
|                     | 2975 (10) | 2977 (10) | 2977 (10) | 2981 (10) | 2983 (10) | 2983 (10) | 3000 (10) | 3007 (10) |
|                     | 3010 (10) | 3042 (10) |           |           |           |           |           |           |
| fHorzCentered       | 1923*(13) | 47-(19)   | 200 (19)  | 353-(19)  | 1040 (19) |           |           |           |
| fHPrint             | 107*(36)  | 259-(37)  | 305 (37)  | 335 (37)  | 338 (37)  | 339 (37)  | 537 (37)  | 888 (37)  |
|                     | 1145 (37) | 1155 (37) | 1189 (37) | 1191 (37) | 1289 (37) | 1298 (37) | 1326 (37) | 1352 (37) |
|                     | 1415 (37) | 1467 (37) | 1488 (37) | 1716 (37) | 1797 (37) | 1808-(37) | 1817-(37) | 1818 (37) |
|                     | 1950 (37) | 1954 (37) | 1978 (37) | 2067 (37) | 2113 (37) | 2167 (37) | 2179 (37) |           |
| fHTE                | 621 ( 6)  | 2592 ( 6) | 2601 ( 6) | 2603 ( 6) | 2622 ( 6) | 2624 ( 6) | 2625 ( 6) | 2639 ( 6) |
|                     | 2640 ( 6) | 2655 ( 6) | 3248 ( 6) | 3272 ( 6) | 3274 ( 6) | 96*(38)   | 374*(38)  | 224 (39)  |
|                     | 27-(40)   | 53 (40)   | 54 (40)   | 56 (40)   | 80-(40)   | 109 (40)  | 110 (40)  | 111 (40)  |
|                     | 178 (40)  | 181 (40)  | 191 (40)  | 206 (40)  | 207 (40)  | 217 (40)  | 219 (40)  | 245 (40)  |
|                     | 247 (40)  | 248 (40)  | 305 (40)  | 313 (40)  | 314 (40)  | 331 (40)  | 332 (40)  | 350 (40)  |
|                     | 419 (40)  | 435 (40)  | 436 (40)  | 438 (40)  | 457 (40)  | 483 (40)  | 484 (40)  | 485 (40)  |
|                     | 534 (40)  | 549 (40)  | 554 (40)  | 555 (40)  | 556 (40)  | 609 (40)  | 611 (40)  | 620 (40)  |
|                     | 657 (40)  | 658 (40)  | 680 (40)  | 682 (40)  | 710 (40)  | 718 (40)  | 729 (40)  | 729 (40)  |
|                     | 853 (40)  | 853 (40)  | 865 (40)  | 865 (40)  | 891 (40)  | 957 (40)  | 966 (40)  | 990 (40)  |
|                     | 992 (40)  | 994 (40)  | 1002 (40) | 1009 (40) | 1018 (40) | 1025 (40) | 1029 (40) | 1040 (40) |

|                  |           |           |           |           |           |           |           |           |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                  | 1132 (40) | 1137 (40) | 1138 (40) | 1139 (40) | 1187 (40) | 1192 (40) | 1233-(40) | 1250 (40) |
|                  | 1266 (40) | 1272 (40) | 1339 (40) | 1340 (40) | 1373 (40) | 1378 (40) | 1379 (40) | 1380 (40) |
|                  | 1411 (40) | 1441 (40) | 1468 (40) | 1493 (40) | 1495 (40) | 1501 (40) | 1522 (40) | 1525 (40) |
|                  | 1543 (40) | 1544 (40) | 1562 (40) | 1607 (40) | 1608 (40) | 1639 (40) | 29-(41)   | 29 (41)   |
|                  | 31 (41)   | 59 (41)   | 77 (41)   | 77 (41)   | 79 (41)   | 112 (41)  | 142 (41)  | 144 (41)  |
|                  | 167 (41)  | 192 (41)  | 193 (41)  | 205 (41)  | 219 (41)  | 231 (41)  | 233 (41)  | 392 (41)  |
|                  | 405 (41)  | 485 (41)  | 556 (41)  | 649 (41)  | 738 (41)  | 830 (41)  | 858 (41)  |           |
| fhTE             | 1350 (40) |           |           |           |           |           |           |           |
| fi               | 16*( 2)   | 29 ( 2)   | 37 ( 2)   | 194*( 2)  | 219 ( 2)  | 222 ( 2)  | 260*( 6)  | 275 ( 6)  |
|                  | 288 ( 6)  | 303*( 6)  | 322 ( 6)  | 333 ( 6)  | 1124*( 6) | 1141 ( 6) | 1144 ( 6) | 1394*( 6) |
|                  | 1411 ( 6) | 1420 ( 6) | 1436*( 6) | 1457 ( 6) | 1467 ( 6) | 1617*( 6) | 1634 ( 6) | 1643 ( 6) |
|                  | 1659*( 6) | 1680 ( 6) | 1689 ( 6) | 1838*( 6) | 1854 ( 6) | 1857 ( 6) | 1873*( 6) | 1890 ( 6) |
|                  | 1893 ( 6) | 2029*( 6) | 2050 ( 6) | 2055 ( 6) | 2083*( 6) | 2106 ( 6) | 2111 ( 6) | 2487*( 6) |
|                  | 2504 ( 6) | 2507 ( 6) | 2673*( 6) | 2691 ( 6) | 2694 ( 6) | 188*( 7)  | 225*( 7)  | 18*( 8)   |
|                  | 297*( 8)  | 300 ( 8)  | 305 ( 8)  | 311 ( 8)  | 243*(12)  | 269 (12)  | 287 (12)  | 835*(14)  |
|                  | 854 (14)  | 888 (14)  | 902*(14)  | 920 (14)  | 965 (14)  | 980*(14)  | 999 (14)  | 1005 (14) |
|                  | 115*(15)  | 134 (15)  | 191 (15)  | 194*(16)  | 226 (16)  | 228 (16)  | 708*(16)  | 724 (16)  |
|                  | 727 (16)  | 1232*(16) | 1252 (16) | 1283 (16) | 1382*(16) | 1414 (16) | 1420 (16) | 1439*(16) |
|                  | 1481 (16) | 1483 (16) | 1494 (16) | 1513 (16) | 1951*(16) | 2005 (16) | 2015 (16) | 2034*(16) |
|                  | 2077 (16) | 2122 (16) | 2133*(16) | 2148 (16) | 2176 (16) | 2190*(16) | 2209 (16) | 2237 (16) |
|                  | 2246*(16) | 2287 (16) | 2293 (16) | 2322*(16) | 2353 (16) | 2380 (16) | 2435*(16) | 2447 (16) |
|                  | 2475 (16) | 2490*(16) | 2506 (16) | 2509 (16) | 14*(17)   | 79 (17)   | 93 (17)   | 303*(17)  |
|                  | 335 (17)  | 337 (17)  | 631*(17)  | 669 (17)  | 709 (17)  | 800*(17)  | 846 (17)  | 873 (17)  |
|                  | 959*(17)  | 983 (17)  | 1006 (17) | 1055*(17) | 1082 (17) | 1178 (17) | 1294*(17) | 1318*(17) |
|                  | 1333 (17) | 1338 (17) | 1354 (17) | 1360 (17) | 1362 (17) | 1372 (17) | 183*(18)  | 206 (18)  |
|                  | 211 (18)  | 426*(18)  | 442 (18)  | 454 (18)  | 22*(19)   | 61 (19)   | 73 (19)   | 91*(19)   |
|                  | 115 (19)  | 149 (19)  | 212*(20)  | 220 (20)  | 247 (20)  | 629*(21)  | 640 (21)  | 645 (21)  |
|                  | 1211*(21) | 1224 (21) | 1234 (21) | 1247*(21) | 1258 (21) | 1263 (21) | 160*(23)  | 209 (23)  |
|                  | 219 (23)  | 248*(37)  | 263 (37)  | 308 (37)  | 626*(37)  | 640 (37)  | 643 (37)  | 1127*(37) |
|                  | 1171 (37) | 1173 (37) | 1429*(37) | 1453*(37) | 1503 (37) | 1522 (37) | 1548 (37) | 1550 (37) |
|                  | 1574*(37) | 1594 (37) | 1608 (37) | 1697*(37) | 1722 (37) | 1734 (37) | 2060*(37) | 2080 (37) |
|                  | 2082 (37) | 2094*(37) | 2109 (37) | 2116 (37) | 1084*(40) | 1109 (40) | 1169 (40) | 1207*(40) |
|                  | 1231 (40) | 1238 (40) | 17*(41)   | 51 (41)   | 84 (41)   | 349*(41)  | 381 (41)  | 385 (41)  |
|                  | 442*(41)  | 472 (41)  | 507 (41)  | 523*(41)  | 634*(41)  | 647 (41)  | 657 (41)  | 819*(41)  |
|                  | 836 (41)  | 853 (41)  |           |           |           |           |           |           |
| fIdentifier      | 462 ( 6)  | 463 ( 6)  | 1339 ( 6) | 1468*(13) | 199-(18)  | 229-(18)  | 250 (18)  | 253 (18)  |
|                  | 958 (18)  | 966 (18)  | 1915 (18) | 55 (19)   |           |           |           |           |
| fIdleFreq        | 2608-( 6) | 2608 ( 6) | 586*(13)  | 15-(15)   | 45 (15)   | 1761 (16) | 1763 (16) | 1768 (16) |
|                  | 58-(40)   | 113-(40)  | 683-(40)  |           |           |           |           |           |
| fieldAddr        | 39*( 1)   | 87*( 1)   | 56*( 2)   | 356*( 2)  | 124*( 5)  | 285*( 5)  | 331*( 5)  | 377*( 5)  |
|                  | 422*( 5)  | 491*( 5)  | 547*( 5)  | 624*( 5)  | 666*( 5)  | 689*( 5)  | 755*( 6)  | 1368*( 6) |
|                  | 1591*( 6) | 1811*( 6) | 2003*( 6) | 2528*( 6) | 2562*( 6) | 2903*( 6) | 3135*( 6) | 3541*( 6) |
|                  | 399*( 9)  | 496*( 9)  | 624*( 9)  | 661*( 9)  | 699*( 9)  | 722*( 9)  | 745*( 9)  | 768*( 9)  |
|                  | 2139*(10) | 2406*(10) | 2762*(10) | 2833*(10) | 3059*(10) | 3094*(10) | 3127*(10) | 3162*(10) |
|                  | 248*(11)  | 621*(12)  | 608*(13)  | 698*(13)  | 1160*(13) | 1732*(13) | 1856*(13) | 2032*(13) |
|                  | 2139*(13) | 2206*(13) | 2271*(13) | 2315*(13) | 2347*(13) | 2467*(13) | 2581*(13) | 40*(15)   |
|                  | 907*(16)  | 1485*(17) | 1903*(18) | 1024*(19) | 295*(20)  | 503*(21)  | 923*(21)  | 1181*(21) |
|                  | 1431*(21) | 120*(22)  | 135*(23)  | 39*(24)   | 357*(25)  | 1096*(26) | 97*(31)   | 115*(31)  |
|                  | 152*(32)  | 162 (32)  | 255*(32)  | 258 (32)  | 290*(32)  | 467*(36)  | 504*(36)  | 875*(37)  |
|                  | 2150*(37) | 357*(38)  | 449*(38)  | 533*(38)  | 587*(38)  | 1634*(40) | 162*(41)  | 611*(41)  |
|                  | 938*(41)  |           |           |           |           |           |           |           |
| FieldClientError | 1131*(37) | 1171 (37) |           |           |           |           |           |           |
| fieldName        | 38*( 1)   | 86*( 1)   | 55*( 2)   | 355*( 2)  | 123*( 5)  | 284*( 5)  | 330*( 5)  | 376*( 5)  |
|                  | 421*( 5)  | 490*( 5)  | 546*( 5)  | 623*( 5)  | 665*( 5)  | 688*( 5)  | 754*( 6)  | 1367*( 6) |
|                  | 1590*( 6) | 1810*( 6) | 2002*( 6) | 2527*( 6) | 2561*( 6) | 2902*( 6) | 3134*( 6) | 3540*( 6) |
|                  | 398*( 9)  | 495*( 9)  | 623*( 9)  | 660*( 9)  | 698*( 9)  | 721*( 9)  | 744*( 9)  | 767*( 9)  |
|                  | 2138*(10) | 2405*(10) | 2761*(10) | 2832*(10) | 3058*(10) | 3093*(10) | 3126*(10) | 3161*(10) |
|                  | 247*(11)  | 620*(12)  | 607*(13)  | 697*(13)  | 1159*(13) | 1731*(13) | 1855*(13) | 2031*(13) |
|                  | 2138*(13) | 2205*(13) | 2270*(13) | 2314*(13) | 2346*(13) | 2466*(13) | 2580*(13) | 39*(15)   |
|                  | 906*(16)  | 1484*(17) | 1902*(18) | 1023*(19) | 294*(20)  | 502*(21)  | 922*(21)  | 1180*(21) |
|                  | 1430*(21) | 119*(22)  | 134*(23)  | 38*(24)   | 356*(25)  | 1095*(26) | 96*(31)   | 115*(31)  |
|                  | 152*(32)  | 163 (32)  | 174 (32)  | 255*(32)  | 258 (32)  | 289*(32)  | 466*(36)  | 503*(36)  |
|                  | 874*(37)  | 2149*(37) | 356*(38)  | 448*(38)  | 532*(38)  | 586*(38)  | 1633*(40) | 161*(41)  |
|                  | 610*(41)  | 937*(41)  |           |           |           |           |           |           |
| Fields           | 38*( 1)   | 86*( 1)   | 55*( 2)   | 62 ( 2)   | 355*( 2)  | 362 ( 2)  | 123*( 5)  | 284*( 5)  |
|                  | 330*( 5)  | 376*( 5)  | 421*( 5)  | 490*( 5)  | 546*( 5)  | 623*( 5)  | 665*( 5)  | 688*( 5)  |
|                  | 754*( 6)  | 766 ( 6)  | 1367*( 6) | 1375 ( 6) | 1590*( 6) | 1598 ( 6) | 1810*( 6) | 1818 ( 6) |
|                  | 2002*( 6) | 2009 ( 6) | 2527*( 6) | 2537 ( 6) | 2561*( 6) | 2567 ( 6) | 2902*( 6) | 2911 ( 6) |
|                  | 3134*( 6) | 3142 ( 6) | 3540*( 6) | 3547 ( 6) | 396*( 9)  | 493*( 9)  | 621*( 9)  | 658*( 9)  |
|                  | 696*( 9)  | 719*( 9)  | 742*( 9)  | 765*( 9)  | 2138*(10) | 2173 (10) | 2405*(10) | 2415 (10) |
|                  | 2761*(10) | 2765 (10) | 2832*(10) | 2840 (10) | 3058*(10) | 3066 (10) | 3093*(10) | 3099 (10) |
|                  | 3126*(10) | 3132 (10) | 3161*(10) | 3168 (10) | 247*(11)  | 620*(12)  | 641 (12)  | 607*(13)  |
|                  | 697*(13)  | 1159*(13) | 1731*(13) | 1855*(13) | 2031*(13) | 2138*(13) | 2205*(13) | 2270*(13) |
|                  | 2314*(13) | 2346*(13) | 2466*(13) | 2580*(13) | 39*(15)   | 47 (15)   | 906*(16)  | 1013 (16) |
|                  | 1484*(17) | 1514 (17) | 1902*(18) | 1919 (18) | 1023*(19) | 1053 (19) | 294*(20)  | 309 (20)  |
|                  | 502*(21)  | 518 (21)  | 922*(21)  | 928 (21)  | 1180*(21) | 1191 (21) | 1430*(21) | 1437 (21) |
|                  | 119*(22)  | 136 (22)  | 134*(23)  | 144 (23)  | 38*(24)   | 47 (24)   | 96*(31)   | 289*(32)  |
|                  | 310 (32)  | 466*(36)  | 503*(36)  | 874*(37)  | 903 (37)  | 2149*(37) | 2156 (37) | 356*(38)  |
|                  | 448*(38)  | 532*(38)  | 586*(38)  | 1633*(40) | 1659 (40) | 161*(41)  | 179 (41)  | 610*(41)  |
|                  | 620 (41)  | 937*(41)  | 944 (41)  |           |           |           |           |           |
|                  | 245*(25)  | 186*(26)  | 162 (32)  |           |           |           |           |           |
| FieldToString    | 40*( 1)   | 88*( 1)   | 57*( 2)   | 357*( 2)  | 124*( 5)  | 285*( 5)  | 331*( 5)  | 378*( 5)  |
| fieldType        | 422*( 5)  | 491*( 5)  | 547*( 5)  | 624*( 5)  | 666*( 5)  | 689*( 5)  | 756*( 6)  | 1369*( 6) |
|                  | 1592*( 6) | 1812*( 6) | 2004*( 6) | 2529*( 6) | 2563*( 6) | 2904*( 6) | 3136*( 6) | 3542*( 6) |
|                  | 400*( 9)  | 497*( 9)  | 625*( 9)  | 662*( 9)  | 700*( 9)  | 723*( 9)  | 746*( 9)  | 769*( 9)  |
|                  | 2140*(10) | 2406*(10) | 2762*(10) | 2833*(10) | 3059*(10) | 3094*(10) | 3127*(10) | 3162*(10) |
|                  | 249*(11)  | 622*(12)  | 609*(13)  | 699*(13)  | 1161*(13) | 1733*(13) | 1857*(13) | 2033*(13) |
|                  | 2140*(13) | 2207*(13) | 2272*(13) | 2316*(13) | 2348*(13) | 2468*(13) | 2582*(13) | 41*(15)   |
|                  | 908*(16)  | 1486*(17) | 1904*(18) | 1025*(19) | 296*(20)  | 504*(21)  | 924*(21)  | 1182*(21) |
|                  | 1432*(21) | 121*(22)  | 136*(23)  | 40*(24)   | 245*(25)  | 350*(25)  | 358*(25)  | 186*(26)  |
|                  | 859*(26)  | 938 (26)  | 1097*(26) | 98*(31)   | 115*(31)  | 152*(32)  | 160 (32)  | 162 (32)  |
|                  | 255*(32)  | 258 (32)  | 291*(32)  | 468*(36)  | 505*(36)  | 876*(37)  | 2151*(37) | 358*(38)  |
|                  | 450*(38)  | 534*(38)  | 588*(38)  | 1635*(40) | 163*(41)  | 612*(41)  | 939*(41)  |           |
|                  | 871*(13)  | 334*(16)  | 349 (16)  | 358 (16)  | 362 (16)  | 416*(16)  | 427 (16)  | 450 (16)  |
|                  | 2927*(16) | 2935-(16) |           |           |           |           |           |           |
| fileFilter       | 893 (17)  | 1232 (17) | 390*(25)  | 172*(26)  | 177-(26)  | 179-(26)  |           |           |
| FileModDate      | 828*(13)  | 1312*(13) | 1317*(13) | 1333*(13) | 270*(16)  | 298 (16)  | 1219*(17) | 1228 (17) |
| fileName         | 1229 (17) | 1232 (17) | 1234 (17) | 1249*(17) | 1263 (17) | 1269 (17) | 1289*(17) | 1365 (17) |
|                  | 1370 (17) |           |           |           |           |           |           |           |

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| filename            | 1257*(13) | 1286*(13) | 571*(17)  | 587 (17)  | 589=(17)  | 591 (17)  | 592=(17)  | 592 (17)  |
|                     | 597=(17)  | 597 (17)  | 903*(17)  | 917=(17)  | 918 (17)  | 928 (17)  | 932=(17)  | 938 (17)  |
|                     | 944 (17)  |           |           |           |           |           |           |           |
| fillCRect           | 1773 ( 6) |           |           |           |           |           |           |           |
| filler              | 102*( 9)  | 1445*(13) | 1881*(13) |           |           |           |           |           |
| filler1             | 2046*(13) | 1218*(16) | 75*(38)   |           |           |           |           |           |
| filler2             | 2052*(13) | 1219*(16) |           |           |           |           |           |           |
| fillInDirID         | 534 (17)  | 393*(25)  | 144 (26)  | 194*(26)  | 197=(26)  | 610 (26)  |           |           |
| fillInDirId         | 243 (26)  |           |           |           |           |           |           |           |
| fillRect            | 1769 ( 6) |           |           |           |           |           |           |           |
| filterProc          | 871*(13)  | 1350*(13) | 2918*(13) | 797*(14)  | 804 (14)  | 336*(16)  | 349 (16)  | 418*(16)  |
|                     | 427 (16)  | 450 (16)  | 2927*(16) | 2937=(16) | 912*(17)  | 918 (17)  | 928 (17)  | 1400*(17) |
|                     | 1420=(17) |           |           |           |           |           |           |           |
| FindColChunk        | 306*( 9)  | 463 (10)  | 877 (10)  | 1163*(10) | 1176=(10) | 1184=(10) | 1231=(10) | 1361 (10) |
|                     | 1550 (10) |           |           |           |           |           |           |           |
| FinderSegProc       | 244*(14)  | 2696 (16) |           |           |           |           |           |           |
| fIndex              | 250*( 5)  | 508*( 5)  | 1138=( 6) | 1142 ( 6) | 1373 ( 6) | 2687=( 6) | 2692 ( 6) | 2908 ( 6) |
| FindLimit           | 445*(37)  | 448=(37)  | 468 (37)  |           |           |           |           |           |
| FindObjID           | 609 (12)  | 130*(31)  | 105*(32)  | 119 (32)  | 125=(32)  | 225 (32)  |           |           |
| FindObjId           | 118=(32)  |           |           |           |           |           |           |           |
| FindPos             | 1073*(14) | 1083=(14) | 1094 (14) | 1099 (14) |           |           |           |           |
| FindRadio           | 1331*( 6) | 1333=( 6) | 1337 ( 6) |           |           |           |           |           |
| FindRowChunk        | 309*( 9)  | 974 (10)  | 1247*(10) | 1260=(10) | 1268=(10) | 1315=(10) | 1392 (10) | 1650 (10) |
|                     | 1817 (10) |           |           |           |           |           |           |           |
| findSeg             | 2683*(16) | 2696=(16) | 2698 (16) | 2703 (16) |           |           |           |           |
| FindStdView         | 2994*(13) | 253*(14)  | 262=(14)  | 263 (14)  | 265=(14)  | 70 (15)   |           |           |
| FindSubview         | 480 ( 6)  | 578 ( 6)  | 704 ( 6)  | 1538*(13) | 1035 (14) | 166 (15)  | 947*(18)  | 981=(18)  |
| FindWindow          | 1565 (16) | 3027 (16) |           |           |           |           |           |           |
| fInfo               | 1251*( 6) | 1264 ( 6) | 1265 ( 6) | 1265 ( 6) | 1265 ( 6) | 1365*(40) | 1408 (40) | 1411 (40) |
| FinishEach          | 248*(12)  | 261 (12)  | 289 (12)  |           |           |           |           |           |
| fInset              | 2197 ( 6) | 2197 ( 6) | 2198 ( 6) | 2198 ( 6) | 2588=( 6) | 2588 ( 6) | 2592 ( 6) | 2592 ( 6) |
|                     | 2622 ( 6) | 2625 ( 6) | 2069*(13) | 114=(21)  | 146=(21)  | 185 (21)  | 239 (21)  | 263 (21)  |
|                     | 383 (21)  | 442=(21)  | 514 (21)  | 783 (21)  | 783 (21)  | 783 (21)  | 783 (21)  | 101*(38)  |
|                     | 31=(40)   | 95=(40)   | 147 (40)  | 283 (40)  | 283 (40)  | 381 (40)  | 381 (40)  | 383 (40)  |
|                     | 383 (40)  | 469 (40)  | 516 (40)  | 518 (40)  | 594 (40)  | 605 (40)  | 629 (40)  | 942 (40)  |
|                     | 1063 (40) | 1220 (40) | 1221 (40) | 1222 (40) | 1268 (40) | 1269 (40) | 1270 (40) | 1642 (40) |
| First               | 95*(11)   | 298*(12)  | 303=(12)  | 305=(12)  | 2161 (16) | 2761 (16) | 744 (17)  | 1463 (18) |
| first               | 120*( 5)  | 591*( 6)  | 597 ( 6)  | 713*( 6)  | 724 ( 6)  | 725=( 6)  | 739=( 6)  | 743 ( 6)  |
|                     | 240*(25)  | 166*(26)  |           |           |           |           |           |           |
| firstChar           | 276*(38)  | 543*(40)  | 554 (40)  |           |           |           |           |           |
| firstCol            | 1087*(10) | 1093=(10) | 1101 (10) | 1110 (10) |           |           |           |           |
| firstEntryThat      | 77*( 1)   | 174 ( 2)  | 181*( 2)  | 184=( 2)  | 251 ( 2)  | 310 ( 2)  |           |           |
| firstHandlerThat    | 804*(13)  | 1021*(16) | 1039=(16) | 1040 (16) | 1044=(16) | 1344 (16) |           |           |
| firstPage           | 1436*(37) | 1471=(37) | 1474 (37) | 1478 (37) | 1498 (37) |           |           |           |
| firstRow            | 1085*(10) | 1092=(10) | 1102 (10) | 1108 (10) |           |           |           |           |
| firstSelectedCell   | 312*( 9)  | 1330*(10) | 1333=(10) |           |           |           |           |           |
| firstSubjobPage     | 1446*(37) | 1505=(37) | 1506 (37) | 1507 (37) | 1514 (37) |           |           |           |
| firstSubviewThat    | 1337 ( 6) | 961 (18)  | 971 (18)  | 988*(18)  | 991=(18)  | 993=(18)  |           |           |
| firstSubviewThat    | 1522*(13) |           |           |           |           |           |           |           |
| firstThat           | 184 ( 2)  | 132*(11)  | 313*(12)  | 329=(12)  | 330 (12)  | 334=(12)  | 745 (12)  | 991 (18)  |
| firstView           | 114*(15)  | 119 (15)  | 127=(15)  | 187=(15)  | 192 (15)  |           |           |           |
| fIsActive           | 1913*(13) | 51=(19)   | 127=(19)  | 260 (19)  | 290=(19)  | 1044 (19) |           |           |
| fIsClosable         | 1915*(13) | 53=(19)   | 129=(19)  | 188 (19)  | 1046 (19) |           |           |           |
| fIsModal            | 1920*(13) | 781=(16)  | 781 (16)  | 1816 (16) | 2901 (16) | 58=(19)   | 132=(19)  | 190 (19)  |
|                     | 435 (19)  | 1051 (19) |           |           |           |           |           |           |
| fIsResizable        | 1914*(13) | 52=(19)   | 128=(19)  | 189 (19)  | 287 (19)  | 460 (19)  | 826 (19)  | 1045 (19) |
| fItemOffset         | 443*( 5)  | 2047=( 6) | 2103=( 6) | 2157 ( 6) | 2197 ( 6) | 2214 ( 6) | 2230 ( 6) | 2536 ( 6) |
| fJust               | 509*( 5)  | 2582 ( 6) | 2688=( 6) | 2717=( 6) | 2747 ( 6) | 2820 ( 6) | 2868=( 6) | 2910 ( 6) |
| fJustification      | 122*(38)  | 37=(40)   | 101=(40)  | 148 (40)  | 1234 (40) | 1351=(40) | 1650 (40) |           |
| fKeepSelecting      | 675*( 9)  | 2964=(10) | 2966=(10) | 2973 (10) | 3064 (10) |           |           |           |
| fKey                | 30*( 1)   | 24 ( 2)   | 31=( 2)   | 32 ( 2)   | 46 ( 2)   | 60 ( 2)   | 76 ( 2)   | 76 ( 2)   |
|                     | 78 ( 2)   | 78 ( 2)   | 147 ( 2)  | 149 ( 2)  | 212 ( 2)  | 259 ( 2)  | 278 ( 2)  | 280 ( 2)  |
|                     | 329 ( 2)  | 331 ( 2)  | 682 ( 6)  | 682 ( 6)  |           |           |           |           |
| fKeyCmdNumber       | 103*(38)  | 32=(40)   | 96=(40)   | 145 (40)  | 1643 (40) |           |           |           |
| flags               | 432=(28)  | 432 (28)  |           |           |           |           |           |           |
| FlashControl        | 53*( 6)   | 480 ( 6)  | 578 ( 6)  |           |           |           |           |           |
| fLastBreak          | 186*(36)  | 278=(37)  | 901 (37)  | 983 (37)  | 989 (37)  | 1001 (37) |           |           |
| fLastCheckedPrinter | 142*(36)  | 286=(37)  | 522 (37)  | 534=(37)  | 892 (37)  |           |           |           |
| fLastCol            | 161*( 9)  | 89=(10)   | 195=(10)  | 918=(10)  | 1179 (10) | 1188 (10) | 1208 (10) | 1234=(10) |
|                     | 2168 (10) |           |           |           |           |           |           |           |
| fLastColChunkNum    | 162*( 9)  | 90=(10)   | 196=(10)  | 919=(10)  | 1181 (10) | 1206 (10) | 1235=(10) | 2169 (10) |
| fLastColIndex       | 164*( 9)  | 92=(10)   | 198=(10)  | 921=(10)  | 1183 (10) | 1208 (10) | 1237=(10) | 2171 (10) |
| fLastColTotal       | 163*( 9)  | 91=(10)   | 197=(10)  | 920=(10)  | 1182 (10) | 1207 (10) | 1236=(10) | 2170 (10) |
| fLastHeight         | 112*(38)  | 34=(40)   | 92=(40)   | 1455=(40) | 1581 (40) | 1584=(40) | 1645 (40) |           |
| fLastIdle           | 589*(13)  | 16=(15)   | 46 (15)   | 1763 (16) | 1766=(16) |           |           |           |
| fLastLine           | 138*(38)  | 47=(40)   | 85=(40)   | 601 (40)  | 625=(40)  | 1657 (40) |           |           |
| fLastPageBreak      | 137*(38)  | 46=(40)   | 84=(40)   | 598 (40)  | 600 (40)  | 624=(40)  | 1658 (40) |           |
| fLastPrinterName    | 147*(36)  | 527 (37)  | 531 (37)  | 893 (37)  |           |           |           |           |
| fLastRow            | 157*( 9)  | 83=(10)   | 189=(10)  | 1015=(10) | 1263 (10) | 1272 (10) | 1292 (10) | 1318=(10) |
|                     | 2163 (10) |           |           |           |           |           |           |           |
| fLastRowChunkNum    | 158*( 9)  | 84=(10)   | 190=(10)  | 1016=(10) | 1265 (10) | 1290 (10) | 1319=(10) | 2164 (10) |
| fLastRowIndex       | 160*( 9)  | 86=(10)   | 192=(10)  | 1018=(10) | 1267 (10) | 1292 (10) | 1321=(10) | 2166 (10) |
| fLastRowTotal       | 159*( 9)  | 85=(10)   | 191=(10)  | 1017=(10) | 1266 (10) | 1291 (10) | 1320=(10) | 2165 (10) |
| fLastStrip          | 184*(36)  | 277 (37)  | 900 (37)  | 982 (37)  | 986 (37)  | 988 (37)  | 1000 (37) |           |
| fLastWholeRect      | 148*( 9)  | 121=(10)  | 170=(10)  | 1998 (10) | 2033=(10) | 2161 (10) |           |           |
| fLineAscent         | 420*( 9)  | 2323 (10) | 2361=(10) | 2410 (10) |           |           |           |           |
| fLineHeight         | 419*( 9)  | 2218 (10) | 2252 (10) | 2360=(10) | 2409 (10) | 2642 (10) |           |           |
| fLocation           | 2593 ( 6) | 1464*(13) | 194=(18)  | 254 (18)  | 561 (18)  | 1037 (18) | 1041 (18) | 1148 (18) |
|                     | 1404 (18) | 1420 (18) | 1440 (18) | 1440 (18) | 1444 (18) | 1445 (18) | 1630 (18) | 1630 (18) |
|                     | 1701 (18) | 1822 (18) | 1910 (18) | 110 (19)  | 110 (19)  | 110 (19)  | 110 (19)  | 111=(19)  |
|                     | 164 (20)  | 225 (20)  | 225 (20)  | 237 (20)  | 237 (20)  | 367 (20)  | 663 (20)  | 203 (39)  |
|                     | 204 (39)  |           |           |           |           |           |           |           |
| fLongMax            | 2228*(13) | 24 (21)   | 1006 (21) | 1036 (21) | 1068 (21) | 1083 (21) | 1111 (21) | 1113=(21) |
|                     | 1145 (21) | 1190 (21) | 1366 (21) |           |           |           |           |           |
| fLongMin            | 2227*(13) | 23 (21)   | 1005 (21) | 1038 (21) | 1092 (21) | 1131 (21) | 1133=(21) | 1145 (21) |
|                     | 1189 (21) |           |           |           |           |           |           |           |
| fLongVal            | 2226*(13) | 23 (21)   | 24 (21)   | 1004 (21) | 1036 (21) | 1038 (21) | 1040 (21) | 1064 (21) |
|                     | 1068=(21) | 1070=(21) | 1071 (21) | 1101 (21) | 1146 (21) | 1148=(21) | 1188 (21) | 1353 (21) |
|                     | 1366=(21) | 1368=(21) | 1372 (21) |           |           |           |           |           |



|                  |            |            |            |            |            |            |            |            |  |
|------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| FlushEvents      | 387 (14)   |            |            |            |            |            |            |            |  |
| FlushVol         | 1066 (17)  | 1198 (17)  |            |            |            |            |            |            |  |
| fMaxChars        | 564* (5)   | 2605* (6)  | 2605 (6)   | 2934* (6)  | 2959* (6)  | 3139 (6)   | 106* (38)  | 33* (40)   |  |
|                  | 97* (40)   | 146 (40)   | 730 (40)   | 854 (40)   | 1644 (40)  |            |            |            |  |
| fmaxChars        | 2986 (6)   |            |            |            |            |            |            |            |  |
| fMaxHorizontal   | 136* (9)   | 95* (10)   | 201* (10)  | 357 (10)   | 881* (10)  | 881 (10)   | 1200 (10)  | 1463 (10)  |  |
|                  | 1600* (10) | 1600 (10)  | 1830 (10)  | 2078 (10)  | 2079 (10)  | 2152 (10)  | 2698* (10) | 2698 (10)  |  |
| fMaximum         | 641* (5)   | 3403* (6)  | 3428* (6)  | 3456 (6)   | 3499 (6)   | 3528 (6)   | 3546 (6)   |            |  |
| fMaxTranslation  | 1760* (13) | 27* (20)   | 199 (20)   | 230 (20)   | 242 (20)   | 266 (20)   | 302 (20)   | 535 (20)   |  |
|                  | 615 (20)   | 617 (20)   |            |            |            |            |            |            |  |
| fMaxVertical     | 137* (9)   | 96* (10)   | 202* (10)  | 356 (10)   | 475 (10)   | 978* (10)  | 978 (10)   | 1284 (10)  |  |
|                  | 1463 (10)  | 1700* (10) | 1700 (10)  | 2081 (10)  | 2082 (10)  | 2153 (10)  |            |            |  |
| fMenuHandle      | 441* (5)   | 2041* (6)  | 2095* (6)  | 2193 (6)   | 2195 (6)   | 2196 (6)   | 2197 (6)   | 2198 (6)   |  |
|                  | 2258 (6)   | 2260 (6)   | 2271 (6)   | 2272 (6)   | 2301 (6)   | 2303 (6)   | 2349 (6)   | 2376 (6)   |  |
|                  | 2430 (6)   | 2431 (6)   | 2442 (6)   | 2447 (6)   | 2448 (6)   | 2449* (6)  | 2464 (6)   | 2467 (6)   |  |
|                  | 2469 (6)   | 2512* (6)  | 2534 (6)   |            |            |            |            |            |  |
| fMenuID          | 440* (5)   | 2060* (6)  | 2116* (6)  | 2263 (6)   | 2313 (6)   | 2318 (6)   | 2325 (6)   | 2445 (6)   |  |
|                  | 2452* (6)  | 2514* (6)  | 2533 (6)   |            |            |            |            |            |  |
| fMenuId          | 2282 (6)   |            |            |            |            |            |            |            |  |
| fMinimalMargins  | 180* (36)  | 284* (37)  | 554 (37)   | 899 (37)   | 1048* (37) | 1050 (37)  | 1836 (37)  |            |  |
| fMinimum         | 640* (5)   | 3402* (6)  | 3427* (6)  | 3455 (6)   | 3499 (6)   | 3528 (6)   | 3545 (6)   |            |  |
| fModDate         | 1107* (13) | 42* (17)   | 252 (17)   | 893* (17)  | 1232* (17) | 1501 (17)  |            |            |  |
| fMode            | 505* (38)  | 532* (41)  | 534* (41)  | 534 (41)   | 544 (41)   | 571* (41)  | 571 (41)   | 615 (41)   |  |
| fMoveBounds      | 1907* (13) | 63* (19)   | 138* (19)  | 790 (19)   | 958 (19)   | 968 (19)   | 970 (19)   | 1033 (19)  |  |
| fmsStr           | 220* (25)  | 50* (26)   | 52 (26)    | 52 (26)    |            |            |            |            |  |
| fName            | 360 (16)   | 460* (16)  | 460 (16)   | 1451 (16)  | 2200 (16)  | 2215 (16)  | 823 (17)   | 826* (17)  |  |
|                  | 831 (17)   | 848 (17)   | 893 (17)   | 932 (17)   | 995* (17)  |            |            |            |  |
| fNewEnd          | 388* (38)  | 112 (39)   | 113 (39)   | 42* (41)   | 127 (41)   | 173 (41)   | 192 (41)   | 486* (41)  |  |
|                  | 650* (41)  | 681* (41)  | 681 (41)   | 780* (41)  | 798* (41)  | 798 (41)   | 801 (41)   | 849 (41)   |  |
| fNewHPrint       | 486* (36)  | 1352 (37)  | 2107* (37) | 2114* (37) | 2115 (37)  | 2128 (37)  | 2129 (37)  | 2155 (37)  |  |
|                  | 2179 (37)  |            |            |            |            |            |            |            |  |
| fNewMouseDown    | 674* (9)   | 2857* (10) | 2888 (10)  | 2960 (10)  | 2967* (10) | 3019* (10) | 3044* (10) | 3063 (10)  |  |
| fNewRgn          | 644* (9)   | 2797* (10) | 2809 (10)  | 2905 (10)  | 2912 (10)  | 2914 (10)  | 2914 (10)  | 2917 (10)  |  |
|                  | 2951 (10)  | 2978 (10)  | 2984 (10)  | 3003 (10)  | 3008 (10)  |            |            |            |  |
| fNewStart        | 387* (38)  | 112 (39)   | 113 (39)   | 41* (41)   | 127 (41)   | 172 (41)   | 192 (41)   | 485* (41)  |  |
|                  | 486 (41)   | 649* (41)  | 650 (41)   | 779* (41)  | 801 (41)   | 849 (41)   |            |            |  |
| fNewStyles       | 391* (38)  | 137 (39)   | 141 (39)   | 144 (39)   | 150 (39)   | 44* (41)   | 98 (41)    | 142 (41)   |  |
|                  | 175 (41)   | 494* (41)  |            |            |            |            |            |            |  |
| fNewText         | 390* (38)  | 43* (41)   | 97 (41)    | 133 (41)   | 137 (41)   | 138 (41)   | 141 (41)   | 142 (41)   |  |
|                  | 144 (41)   | 144 (41)   | 146 (41)   | 174 (41)   | 190 (41)   | 484* (41)  | 652* (41)  | 653 (41)   |  |
|                  | 680 (41)   | 801 (41)   | 868 (41)   | 868 (41)   |            |            |            |            |  |
| fNewTextStyle    | 507* (38)  | 530* (41)  | 570 (41)   | 618 (41)   |            |            |            |            |  |
| fNextHandler     | 585* (13)  | 14* (15)   | 27 (15)    | 30 (15)    | 44 (15)    | 64 (15)    | 65 (15)    | 123 (15)   |  |
|                  | 124 (15)   | 203 (15)   | 204 (15)   | 235 (15)   | 236 (15)   | 247 (15)   | 248 (15)   | 264 (15)   |  |
|                  | 265 (15)   | 278 (15)   | 279 (15)   | 289 (15)   | 290 (15)   | 303 (15)   | 304 (15)   | 325 (15)   |  |
|                  | 326 (15)   | 895 (16)   | 1036 (16)  | 1841* (16) | 1853 (16)  | 1855* (16) | 1855 (16)  | 1861 (16)  |  |
|                  | 357 (18)   | 358* (18)  | 366 (18)   | 367* (18)  | 1541 (18)  | 1542* (18) | 733* (19)  | 738* (19)  |  |
| fnfErr           | 550 (17)   | 648 (17)   | 945 (17)   | 966 (17)   | 1168 (17)  | 1265 (17)  | 1306 (17)  | 1367 (17)  |  |
|                  | 1213 (37)  |            |            |            |            |            |            |            |  |
| fNumOfColChunks  | 156* (9)   | 93* (10)   | 199* (10)  | 455 (10)   | 725 (10)   | 730 (10)   | 781 (10)   | 798 (10)   |  |
|                  | 893* (10)  | 893 (10)   | 904* (10)  | 904 (10)   | 1171 (10)  | 1198 (10)  | 1359 (10)  | 1542 (10)  |  |
|                  | 1543 (10)  | 1545 (10)  | 1546 (10)  | 1566 (10)  | 1578* (10) | 1578 (10)  | 1595* (10) | 1595 (10)  |  |
|                  | 1894 (10)  | 2084 (10)  | 2167 (10)  |            |            |            |            |            |  |
| fNumOfCols       | 138* (9)   | 51* (10)   | 151* (10)  | 254 (10)   | 259 (10)   | 317 (10)   | 401 (10)   | 405 (10)   |  |
|                  | 441 (10)   | 445 (10)   | 497 (10)   | 517 (10)   | 700 (10)   | 853 (10)   | 859 (10)   | 876 (10)   |  |
|                  | 915* (10)  | 915 (10)   | 968 (10)   | 1059 (10)  | 1095 (10)  | 1171 (10)  | 1196 (10)  | 1202 (10)  |  |
|                  | 1350 (10)  | 1353 (10)  | 1533 (10)  | 1542 (10)  | 1562 (10)  | 1564 (10)  | 1599* (10) | 1599 (10)  |  |
|                  | 1603 (10)  | 1717 (10)  | 1885 (10)  | 1888 (10)  | 2150 (10)  | 2221 (10)  | 2223 (10)  | 2254 (10)  |  |
|                  | 2256 (10)  | 2696 (10)  | 2945 (10)  |            |            |            |            |            |  |
| fNumOfRowChunks  | 155* (9)   | 87* (10)   | 193* (10)  | 744 (10)   | 749 (10)   | 783 (10)   | 789 (10)   | 990* (10)  |  |
|                  | 990 (10)   | 1001* (10) | 1001 (10)  | 1255 (10)  | 1282 (10)  | 1390 (10)  | 1642 (10)  | 1643 (10)  |  |
|                  | 1645 (10)  | 1646 (10)  | 1666 (10)  | 1678* (10) | 1678 (10)  | 1695* (10) | 1695 (10)  | 1809 (10)  |  |
|                  | 1920 (10)  | 2107 (10)  | 2162 (10)  |            |            |            |            |            |  |
| fNumOfRows       | 139* (9)   | 50* (10)   | 150* (10)  | 253 (10)   | 255 (10)   | 318 (10)   | 401 (10)   | 406 (10)   |  |
|                  | 497 (10)   | 518 (10)   | 700 (10)   | 870 (10)   | 950 (10)   | 956 (10)   | 973 (10)   | 1012* (10) |  |
|                  | 1012 (10)  | 1070 (10)  | 1094 (10)  | 1255 (10)  | 1280 (10)  | 1286 (10)  | 1381 (10)  | 1384 (10)  |  |
|                  | 1633 (10)  | 1642 (10)  | 1662 (10)  | 1664 (10)  | 1699* (10) | 1699 (10)  | 1703 (10)  | 1729 (10)  |  |
|                  | 1795 (10)  | 1799 (10)  | 1911 (10)  | 1914 (10)  | 2148 (10)  | 2251 (10)  | 2252 (10)  | 2487 (10)  |  |
|                  | 2548 (10)  | 2668 (10)  | 2945 (10)  |            |            |            |            |            |  |
| fObjPtr          | 63* (11)   | 67* (12)   | 122 (12)   | 123 (12)   | 439 (12)   | 440 (12)   | 609* (12)  | 634 (12)   |  |
| Focus            | 2654 (6)   | 2888 (6)   | 3247 (6)   | 3271 (6)   | 3295 (6)   | 482* (9)   | 2334* (10) | 2336 (10)  |  |
|                  | 2339* (10) | 2342* (10) | 1582* (13) | 1795* (13) | 1973* (13) | 2104* (13) | 1608 (16)  | 3128 (16)  |  |
|                  | 317 (18)   | 904 (18)   | 1000* (18) | 1011* (18) | 1014 (18)  | 1020 (18)  | 1050 (18)  | 1057* (18) |  |
|                  | 1067 (18)  | 1287 (18)  | 1331 (18)  | 1344 (18)  | 1360 (18)  | 1770 (18)  | 460 (19)   | 502* (19)  |  |
|                  | 515* (19)  | 524* (19)  | 527* (19)  | 538 (19)   | 827 (19)   | 1008 (19)  | 317* (20)  | 320* (20)  |  |
|                  | 320 (20)   | 407 (20)   | 486 (20)   | 337* (21)  | 339 (21)   | 342* (21)  | 345* (21)  | 369 (21)   |  |
|                  | 899 (21)   | 87 (23)    | 218 (23)   | 223 (39)   | 680 (40)   | 704 (40)   | 863 (40)   | 886 (40)   |  |
|                  | 1181 (40)  | 1304 (40)  | 1332 (40)  | 1370 (40)  | 265 (41)   | 281 (41)   | 301 (41)   | 363 (41)   |  |
|                  | 554 (41)   |            |            |            |            |            |            |            |  |
| focus            | 570 (10)   |            |            |            |            |            |            |            |  |
| FocusOnBorder    | 312* (36)  | 912* (37)  | 1605 (37)  |            |            |            |            |            |  |
| FocusOnInterior  | 2351* (13) | 1018 (18)  | 110* (24)  | 315* (36)  | 929* (37)  | 1597 (37)  |            |            |  |
| FocusOnSuperView | 1584* (13) | 1975* (13) | 1023 (18)  | 1064* (18) | 1067* (18) | 1069* (18) | 1576 (18)  | 535* (19)  |  |
|                  | 538* (19)  |            |            |            |            |            |            |            |  |
| fOldCell         | 673* (9)   | 2943 (10)  | 3018* (10) | 3037 (10)  | 3040 (10)  | 3062 (10)  |            |            |  |
| fOldEnd          | 378* (38)  | 111 (39)   | 113 (39)   | 34* (41)   | 62 (41)    | 111 (41)   | 169 (41)   | 205 (41)   |  |
|                  | 220 (41)   | 487 (41)   | 544 (41)   | 556 (41)   | 781 (41)   |            |            |            |  |
| fOldHPrint       | 485* (36)  | 2106* (37) | 2110* (37) | 2111 (37)  | 2113 (37)  | 2126 (37)  | 2127 (37)  | 2154 (37)  |  |
|                  | 2167 (37)  |            |            |            |            |            |            |            |  |
| fOldRgn          | 643* (9)   | 2795* (10) | 2808 (10)  | 2908 (10)  | 2911 (10)  | 2912 (10)  | 2912 (10)  | 2913 (10)  |  |
|                  | 3000 (10)  | 3003 (10)  | 3003 (10)  | 3004 (10)  |            |            |            |            |  |
| fOldStart        | 377* (38)  | 111 (39)   | 113 (39)   | 33* (41)   | 59 (41)    | 62 (41)    | 111 (41)   | 168 (41)   |  |
|                  | 205 (41)   | 219 (41)   | 219 (41)   | 220 (41)   | 487 (41)   | 544 (41)   | 556 (41)   | 778* (41)  |  |
|                  | 781 (41)   | 845 (41)   |            |            |            |            |            |            |  |
| fOldStyles       | 385* (38)  | 120 (39)   | 124 (39)   | 127 (39)   | 133 (39)   | 39* (41)   | 79* (41)   | 80 (41)    |  |
|                  | 81 (41)    | 96 (41)    | 171 (41)   | 231 (41)   | 391 (41)   | 588 (41)   | 741 (41)   | 747 (41)   |  |
|                  | 748 (41)   | 754 (41)   | 757 (41)   | 760 (41)   | 761 (41)   | 762 (41)   | 771 (41)   | 917 (41)   |  |
| fOldText         | 380* (38)  | 38* (41)   | 61* (41)   | 95 (41)    | 170 (41)   | 226 (41)   | 227 (41)   | 230 (41)   |  |
|                  | 233 (41)   | 235 (41)   | 336* (41)  | 386 (41)   | 775 (41)   | 777 (41)   | 866 (41)   | 866 (41)   |  |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| fOldTextStyle     | 506*(38)  | 529=(41)  | 586 (41)  | 617 (41)  |           |           |           |           |  |
| Font2Script       | 1186 (40) | 1393 (40) |           |           |           |           |           |           |  |
| fontAscent        | 409*(40)  | 436 (40)  | 1416=(40) |           |           |           |           |           |  |
| fontHt            | 1247*( 6) | 1265=( 6) | 1270 ( 6) | 1285 ( 6) |           |           |           |           |  |
| FontInfo          | 1251 ( 6) | 2353 (10) | 301 (40)  | 416 (40)  | 1365 (40) | 1555 (40) |           |           |  |
| fontName          | 252*(25)  | 264*(25)  | 253*(26)  | 259 (26)  | 260 (26)  | 262 (26)  | 265 (26)  | 357*(26)  |  |
|                   | 368=(26)  | 376=(26)  | 381 (26)  |           |           |           |           |           |  |
| fontNum           | 264*(25)  | 256*(26)  | 261=(26)  | 263=(26)  | 265 (26)  | 266 (26)  | 357*(26)  | 363 (26)  |  |
|                   | 365 (26)  | 366 (26)  | 372 (26)  | 373 (26)  | 374 (26)  | 381 (26)  |           |           |  |
| fontNumber        | 33*(14)   | 519 (14)  |           |           |           |           |           |           |  |
| fontSize          | 34*(14)   | 485*(14)  | 519 (14)  | 679=(14)  | 683=(14)  | 686=(14)  | 687 (14)  | 264*(25)  |  |
|                   | 357*(26)  | 369 (26)  | 377 (26)  |           |           |           |           |           |  |
| fOpenInitially    | 1919*(13) | 1451 (17) | 57=(19)   | 137=(19)  | 195 (19)  | 1050 (19) |           |           |  |
| ForAllDocumentsDo | 907*(13)  | 311 (16)  | 529 (16)  | 1051*(16) | 1069 (16) |           |           |           |  |
| ForAllViewsDo     | 1421*(13) | 436*(17)  | 1437 (17) | 573 (37)  |           |           |           |           |  |
| ForAllWindowsDo   | 910*(13)  | 1426*(13) | 1060*(16) | 1066 (16) | 212 (17)  | 454*(17)  | 1391 (17) | 1455 (17) |  |
|                   | 418 (19)  |           |           |           |           |           |           |           |  |
| forceBusy         | 79*( 3)   | 88*( 4)   | 96 ( 4)   |           |           |           |           |           |  |
| forced            | 1891*(13) |           |           |           |           |           |           |           |  |
| forceOn           | 250 ( 8)  | 696 (26)  | 715 (26)  |           |           |           |           |           |  |
| ForceOnScreen     | 2010*(13) | 158 (19)  | 546*(19)  |           |           |           |           |           |  |
| ForceRedraw       | 1359 ( 6) | 1582 ( 6) | 1802 ( 6) | 1994 ( 6) | 2519 ( 6) | 2870 ( 6) | 1602*(13) | 779 (16)  |  |
|                   | 2769 (16) | 1076*(18) | 1443 (18) | 1448 (18) | 1467 (18) | 1482 (18) | 1654 (18) | 1675 (18) |  |
|                   | 1786 (18) | 629 (20)  | 431 (21)  | 301 (23)  | 1084 (37) | 1280 (40) | 1353 (40) | 1428 (40) |  |
| forceUnchanged    | 250 ( 8)  | 696 (26)  | 715 (26)  |           |           |           |           |           |  |
| forCode           | 337*(27)  | 991*(28)  | 995 (28)  |           |           |           |           |           |  |
| ForeColor         | 808 (26)  |           |           |           |           |           |           |           |  |
| forOther          | 337*(27)  | 991*(28)  | 996 (28)  |           |           |           |           |           |  |
| forPrinting       | 1174*(13) | 1180*(13) | 1214*(13) | 275*(17)  | 366*(17)  | 799*(17)  | 869 (17)  |           |  |
| foundCurrent      | 716*( 6)  | 728=( 6)  | 729 ( 6)  | 731 ( 6)  | 736=( 6)  |           |           |           |  |
| foundView         | 949*(18)  | 959=(18)  | 962 (18)  | 967=(18)  | 970=(18)  | 974 (18)  | 981 (18)  |           |  |
| fPadding          | 393*(38)  | 115 (39)  | 46=(41)   | 63=(41)   | 64 (41)   | 99 (41)   | 113 (41)  | 176 (41)  |  |
|                   | 195 (41)  | 501 (41)  | 684 (41)  | 775 (41)  | 796 (41)  |           |           |           |  |
| fPageAreas        | 94*(36)   | 281 (37)  | 368 (37)  | 372 (37)  | 376 (37)  | 382 (37)  | 483 (37)  | 515 (37)  |  |
|                   | 539 (37)  | 541 (37)  | 551 (37)  | 564 (37)  | 565 (37)  | 880 (37)  | 881 (37)  | 882 (37)  |  |
|                   | 883 (37)  | 916 (37)  | 917 (37)  | 937 (37)  | 1052 (37) | 1060 (37) | 1065 (37) | 1066 (37) |  |
|                   | 1095 (37) | 1634 (37) | 1644 (37) | 1835 (37) | 1870 (37) | 1886 (37) | 1896 (37) | 1915 (37) |  |
| fPageDirection    | 150*(36)  | 266=(37)  | 894 (37)  | 1233 (37) | 1236 (37) | 2038 (37) | 2039 (37) |           |  |
| fPageStrips       | 115*(36)  | 798 (37)  | 889 (37)  | 1109 (37) | 1235 (37) | 1237 (37) | 1469=(37) | 1666=(37) |  |
|                   | 2039 (37) |           |           |           |           |           |           |           |  |
| fParamTxt         | 59*( 5)   | 271=( 6)  | 279=( 6)  | 314=( 6)  | 326=( 6)  | 382 ( 6)  | 635 ( 6)  | 690 ( 6)  |  |
|                   | 761 ( 6)  |           |           |           |           |           |           |           |  |
| fPenSize          | 1261 ( 6) | 1261 ( 6) | 1269 ( 6) | 1269 ( 6) | 2938=( 6) | 3123 ( 6) | 2068*(13) | 112=(21)  |  |
|                   | 147=(21)  | 180 (21)  | 267 (21)  | 325 (21)  | 513 (21)  | 713 (21)  |           |           |  |
| fPreDocName       | 1037 (19) |           |           |           |           |           |           |           |  |
| fPreDocname       | 1911*(13) | 2165 (16) | 68=(19)   | 144=(19)  | 911 (19)  | 915 (19)  |           |           |  |
| fPreferColor      | 301*( 5)  | 347*( 5)  | 1407*( 6) | 1412 ( 6) | 1416=( 6) | 1452=( 6) | 1458 ( 6) | 1462=( 6) |  |
|                   | 1493 ( 6) | 1545 ( 6) | 1566 ( 6) | 1595 ( 6) | 1630=( 6) | 1635 ( 6) | 1639=( 6) | 1675=( 6) |  |
|                   | 1681 ( 6) | 1685*( 6) | 1715 ( 6) | 1761 ( 6) | 1765 ( 6) | 1786 ( 6) | 1815 ( 6) |           |  |
| fPrinterDev       | 138*(36)  | 271=(37)  | 516 (37)  | 549=(37)  | 891 (37)  |           |           |           |  |
| fPrintExtent      | 97*(36)   | 417 (37)  | 423 (37)  | 461 (37)  | 848 (37)  | 885 (37)  | 980 (37)  | 981 (37)  |  |
|                   | 994 (37)  | 1262 (37) | 1782=(37) |           |           |           |           |           |  |
| fPrintHandler     | 1471*(13) | 978 (17)  | 979 (17)  | 202=(18)  | 296 (18)  | 297 (18)  | 525=(18)  | 544 (18)  |  |
|                   | 643 (18)  | 644 (18)  | 651 (18)  | 653 (18)  | 710 (18)  | 719 (18)  | 730 (18)  | 739 (18)  |  |
|                   | 758 (18)  | 759 (18)  | 769 (18)  | 789 (18)  | 790 (18)  | 814 (18)  | 815 (18)  | 824 (18)  |  |
|                   | 825 (18)  | 834 (18)  | 855 (18)  | 856 (18)  | 915 (18)  | 1918 (18) | 324 (37)  | 588 (40)  |  |
| fPrintInfo        | 1100*(13) | 27=(17)   | 34=(17)   | 114 (17)  | 115 (17)  | 372 (17)  | 374=(17)  | 375 (17)  |  |
|                   | 379 (17)  | 415 (17)  | 424 (17)  | 1496 (17) | 303 (37)  | 305=(37)  | 335 (37)  | 1806 (37) |  |
|                   | 1808 (37) |           |           |           |           |           |           |           |  |
| fProcID           | 1906*(13) | 43=(19)   | 117=(19)  | 187 (19)  | 879 (19)  | 1032 (19) |           |           |  |
| FrameOval         | 406 (18)  |           |           |           |           |           |           |           |  |
| FrameRect         | 1272 ( 6) | 2407 ( 6) | 495 (18)  | 1394 (21) | 88 (22)   | 378 (37)  | 383 (37)  |           |  |
| FrameRgn          | 591 (10)  | 596 (10)  | 604 (10)  |           |           |           |           |           |  |
| FrameRoundRect    | 414 (18)  |           |           |           |           |           |           |           |  |
| frameSize         | 556*(20)  | 568=(20)  | 570=(20)  | 570 (20)  | 570 (20)  | 571 (20)  |           |           |  |
| Free              | 35*( 1)   | 65*( 1)   | 25 ( 2)   | 43*( 2)   | 48 ( 2)   | 109*( 2)  | 117 ( 2)  | 122 ( 2)  |  |
|                   | 123 ( 2)  | 82*( 5)   | 269*( 5)  | 321*( 5)  | 367*( 5)  | 412*( 5)  | 461*( 5)  | 528*( 5)  |  |
|                   | 267 ( 6)  | 310 ( 6)  | 379*( 6)  | 384 ( 6)  | 1131 ( 6) | 1212*( 6) | 1215 ( 6) | 1401 ( 6) |  |
|                   | 1443 ( 6) | 1521*( 6) | 1524 ( 6) | 1624 ( 6) | 1666 ( 6) | 1740*( 6) | 1743 ( 6) | 1845 ( 6) |  |
|                   | 1880 ( 6) | 1940*( 6) | 1943 ( 6) | 2037 ( 6) | 2091 ( 6) | 2177*( 6) | 2180 ( 6) | 2495 ( 6) |  |
|                   | 2680 ( 6) | 2769*( 6) | 2772 ( 6) | 206*( 9)  | 650*( 9)  | 290*(10)  | 300 (10)  | 2806*(10) |  |
|                   | 2811 (10) | 347 (12)  | 365 (12)  | 370 (12)  | 594*(13)  | 1143*(13) | 1488*(13) | 1785*(13) |  |
|                   | 1945*(13) | 2166*(13) | 2300*(13) | 2568*(13) | 845 (14)  | 912 (14)  | 23*(15)   | 31 (15)   |  |
|                   | 508 (16)  | 568 (16)  | 2263 (16) | 2310 (16) | 2477 (16) | 3235 (16) | 21 (17)   | 101*(17)  |  |
|                   | 117 (17)  | 214 (17)  | 494 (17)  | 187 (18)  | 286*(18)  | 297 (18)  | 299 (18)  | 1095 (18) |  |
|                   | 29 (19)   | 98 (19)   | 226*(19)  | 241 (19)  | 388 (19)  | 122*(20)  | 126 (20)  | 128 (20)  |  |
|                   | 130 (20)  | 216 (20)  | 590*(21)  | 595 (21)  | 633 (21)  | 1215 (21) | 1251 (21) | 1296*(21) |  |
|                   | 1312 (21) | 63*(23)   | 78*(31)   | 250*(33)  | 141 (32)  | 201*(36)  | 493*(36)  | 255 (37)  |  |
|                   | 316*(37)  | 340 (37)  | 1358 (37) | 2101 (37) | 2124*(37) | 2130 (37) | 170*(38)  | 403*(38)  |  |
|                   | 469*(38)  | 557*(38)  | 176*(40)  | 194 (40)  | 1211 (40) | 24 (41)   | 93*(41)   | 101 (41)  |  |
|                   | 333*(41)  | 337 (41)  | 359 (41)  | 452 (41)  | 641 (41)  | 665*(41)  | 669 (41)  |           |  |
|                   | 221*(11)  | 342*(12)  |           |           |           |           |           |           |  |
| FreeAll           | 1049*(17) | 1121=(17) | 1131 (17) | 1152 (17) |           |           |           |           |  |
| freeBlks          | 1148*(13) | 464*(17)  | 991 (17)  |           |           |           |           |           |  |
| FreeData          | 114*( 2)  | 121 ( 2)  |           |           |           |           |           |           |  |
| FreeEntry         | 1243*(13) | 106 (17)  | 473*(17)  | 1260 (17) | 1358 (17) |           |           |           |  |
| FreeFile          | 1153*(13) | 1490*(13) | 181 (16)  | 386 (16)  | 491*(17)  | 1089*(18) | 1093 (18) |           |  |
| FreeFromClipboard | 506*(16)  | 532 (16)  |           |           |           |           |           |           |  |
| FreeIt            | 73*(11)   | 360*(12)  | 109 (17)  | 112 (17)  | 290 (18)  |           |           |           |  |
| FreeList          | 382 ( 6)  | 383 ( 6)  | 344*(12)  | 351 (12)  | 362*(12)  | 369 (12)  | 119 (15)  | 2142 (16) |  |
| FreeObject        | 2197 (16) | 2442 (16) | 1311 (21) | 106*(21)  | 137*(32)  |           |           |           |  |
|                   | 1872*(13) | 1895*(13) | 134 (19)  | 192=(19)  |           |           |           |           |  |
| freeOnClosing     | 2847*(13) | 273*(14)  | 991 (14)  | 1291 (16) | 245 (19)  |           |           |           |  |
| FreeWmgrWindow    | 1108*(13) | 39=(17)   | 741 (17)  | 1502 (17) |           |           |           |           |  |
| fReopenAlert      | 1908*(13) | 851 (14)  | 323 (19)  | 567 (19)  | 568 (19)  | 570 (19)  | 571 (19)  | 855 (19)  |  |
| fResizeLimits     | 876 (19)  | 877 (19)  | 1034 (19) |           |           |           |           |           |  |
|                   | 55*(28)   | 223*(28)  | 256 (28)  |           |           |           |           |           |  |
| fromGZ            | 220*( 9)  | 248*( 9)  | 542*(10)  | 553 (10)  | 563*(10)  | 574 (10)  | 575=(10)  | 581 (10)  |  |
| fromHL            | 585 (10)  | 601 (10)  | 1594*(13) | 2982*(13) | 1205*(14) | 1212 (14) | 1215 (14) | 1221 (14) |  |

|                    |           |           |           |           |           |           |           |           |  |
|--------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| frontmost          | 776*(18)  |           |           |           |           |           |           |           |  |
| FrontWindow        | 841*(16)  | 863-(16)  | 864 (16)  | 865 (16)  |           |           |           |           |  |
| FrontWindow        | 743 (16)  |           |           |           |           |           |           |           |  |
|                    | 523 (16)  | 740 (16)  | 850 (16)  | 863 (16)  | 1410 (16) | 1569 (16) | 1585 (16) | 1680 (16) |  |
|                    | 2409 (16) | 697 (19)  | 73 (37)   |           |           |           |           |           |  |
| fRowHeights        | 141*( 9)  | 79-(10)   | 185-(10)  | 298 (10)  | 745 (10)  | 978 (10)  | 980 (10)  | 980 (10)  |  |
|                    | 982 (10)  | 984 (10)  | 988 (10)  | 988 (10)  | 993 (10)  | 993 (10)  | 995 (10)  | 995 (10)  |  |
|                    | 996 (10)  | 999 (10)  | 999 (10)  | 1274 (10) | 1283 (10) | 1285 (10) | 1297 (10) | 1299 (10) |  |
|                    | 1300 (10) | 1309 (10) | 1310 (10) | 1391 (10) | 1393 (10) | 1643 (10) | 1645 (10) | 1646 (10) |  |
|                    | 1651 (10) | 1653 (10) | 1653 (10) | 1659 (10) | 1670 (10) | 1673 (10) | 1676 (10) | 1677 (10) |  |
|                    | 1684 (10) | 1687 (10) | 1690 (10) | 1690 (10) | 1691 (10) | 1691 (10) | 1692 (10) | 1693 (10) |  |
|                    | 1694 (10) | 1812 (10) | 1813 (10) | 1819 (10) | 1920 (10) | 2109 (10) | 2112 (10) | 2122 (10) |  |
|                    | 2122 (10) | 2123 (10) | 2126 (10) | 2127 (10) | 2149 (10) |           |           |           |  |
| fRowInset          | 144*( 9)  | 58-(10)   | 60-(10)   | 158-(10)  | 160-(10)  | 263 (10)  | 778 (10)  | 790 (10)  |  |
|                    | 792 (10)  | 811 (10)  | 1459 (10) | 1486 (10) | 2154 (10) | 2252 (10) | 2642 (10) |           |  |
| fRsrcID            | 249*( 5)  | 302*( 5)  | 348*( 5)  | 393*( 5)  | 439*( 5)  | 507*( 5)  | 1137-( 6) | 1139 ( 6) |  |
|                    | 1142 ( 6) | 1316-( 6) | 1372 ( 6) | 1408-( 6) | 1409 ( 6) | 1413 ( 6) | 1417 ( 6) | 1455 ( 6) |  |
|                    | 1459 ( 6) | 1463 ( 6) | 1495 ( 6) | 1498 ( 6) | 1541 ( 6) | 1565-( 6) | 1596 ( 6) | 1631-( 6) |  |
|                    | 1632 ( 6) | 1636 ( 6) | 1640 ( 6) | 1676-( 6) | 1678 ( 6) | 1682 ( 6) | 1686 ( 6) | 1717 ( 6) |  |
|                    | 1720 ( 6) | 1760 ( 6) | 1785-( 6) | 1816 ( 6) | 1851-( 6) | 1852 ( 6) | 1855 ( 6) | 1887-( 6) |  |
|                    | 1888 ( 6) | 1891 ( 6) | 1918 ( 6) | 1921 ( 6) | 1959 ( 6) | 1981-( 6) | 2007 ( 6) | 2059-( 6) |  |
|                    | 2115-( 6) | 2152 ( 6) | 2155 ( 6) | 2267 ( 6) | 2268 ( 6) | 2451-( 6) | 2513-( 6) | 2532 ( 6) |  |
|                    | 2686-( 6) | 2689 ( 6) | 2692 ( 6) | 2858-( 6) | 2907 ( 6) |           |           |           |  |
| frsrcID            | 1453-( 6) |           |           |           |           |           |           |           |  |
| fRsrcOpen          | 1123*(13) | 50-(17)   | 60 (17)   | 476 (17)  | 840 (17)  | 863 (17)  | 883 (17)  | 1235 (17) |  |
|                    | 1255 (17) | 1335 (17) | 1508 (17) |           |           |           |           |           |  |
| fRsrcPerm          | 1127*(13) | 75-(17)   | 850 (17)  | 1236 (17) | 1336 (17) | 1510 (17) |           |           |  |
| fRsrcRefNum        | 865 (17)  | 1512 (17) |           |           |           |           |           |           |  |
| fRsrcRefNum        | 1131*(13) | 31-(17)   | 478 (17)  | 885-(17)  | 889-(17)  | 1239-(17) | 1340-(17) |           |  |
| fSavedTEHandle     | 99*(38)   | 30-(40)   | 83-(40)   | 180 (40)  | 183 (40)  | 192-(40)  | 1527 (40) | 1529 (40) |  |
|                    | 1530 (40) | 1531-(40) | 1641 (40) |           |           |           |           |           |  |
| fSaveExists        | 1110*(13) | 291 (16)  | 40-(17)   | 205 (17)  | 246 (17)  | 325 (17)  | 521 (17)  | 853-(17)  |  |
|                    | 993 (17)  | 1226-(17) | 1331 (17) | 1503 (17) |           |           |           |           |  |
| fSaveInPlace       | 1136*(13) | 70-(17)   | 72-(17)   | 1140 (17) | 1155 (17) | 1513 (17) |           |           |  |
| fSavePrintInfo     | 1090*(13) | 35-(17)   | 354 (17)  | 370 (17)  | 413 (17)  | 1494 (17) |           |           |  |
| fSBarOffsets       | 1764*(13) | 942-(14)  | 28-(20)   | 66 (20)   | 101 (20)  | 166 (20)  | 168 (20)  | 169 (20)  |  |
|                    | 225 (20)  | 226 (20)  | 226 (20)  | 237 (20)  | 238 (20)  | 238 (20)  | 306 (20)  |           |  |
| FSClose            | 34 (26)   | 628 (26)  |           |           |           |           |           |           |  |
| fScrapCount        | 2554*(13) | 29-(23)   | 51-(23)   | 142 (23)  | 191 (23)  | 266 (23)  | 271-(23)  |           |  |
| fScrollbar         | 1761*(13) | 21-(20)   | 22-(20)   | 52-(20)   | 53-(20)   | 93 (20)   | 94 (20)   | 125 (20)  |  |
|                    | 126 (20)  | 127 (20)  | 128 (20)  | 160 (20)  | 233-(20)  | 245-(20)  | 303 (20)  | 304 (20)  |  |
|                    | 357-(20)  | 407 (20)  | 407 (20)  | 410 (20)  | 421 (20)  | 465 (20)  | 466 (20)  | 467 (20)  |  |
|                    | 468 (20)  | 618 (20)  | 619 (20)  | 1304 (21) | 195 (39)  | 195 (39)  |           |           |  |
| fScrollbar         | 407 (20)  |           |           |           |           |           |           |           |  |
| fScroller          | 2458*(13) | 3180 (16) | 26-(22)   | 36 (22)   | 133 (22)  | 136*(38)  | 193 (39)  | 28-(40)   |  |
|                    | 60 (40)   | 61 (40)   | 81-(40)   | 244-(40)  | 1304 (40) | 1656 (40) |           |           |  |
| fScrollers         | 2281*(13) | 1219-(21) | 1225-(21) | 1227 (21) | 1231 (21) | 1259-(21) | 1261 (21) | 1309 (21) |  |
|                    | 1310 (21) | 1311 (21) | 1369 (21) | 1420 (21) | 1436 (21) |           |           |           |  |
| fScrollLimit       | 1759*(13) | 95 (20)   | 96 (20)   | 301 (20)  | 417 (20)  | 610-(20)  |           |           |  |
| fScrollUnit        | 1762*(13) | 97 (20)   | 98 (20)   | 196 (20)  | 198 (20)  | 305 (20)  | 536 (20)  | 536 (20)  |  |
|                    | 537 (20)  | 559 (20)  | 641 (20)  | 642 (20)  |           |           |           |           |  |
| fSelections        | 146*( 9)  | 105-(10)  | 205-(10)  | 292 (10)  | 873 (10)  | 873 (10)  | 970 (10)  | 970 (10)  |  |
|                    | 1127 (10) | 1333 (10) | 1427 (10) | 1764 (10) | 1775 (10) | 1860 (10) | 1862 (10) | 1863 (10) |  |
|                    | 1864 (10) | 1998 (10) | 2023 (10) | 2023 (10) | 2025 (10) | 2029 (10) | 2029 (10) | 2032 (10) |  |
|                    | 2143 (10) | 2580 (10) |           |           |           |           |           |           |  |
| FSFCBLen           | 80*(26)   | 119 (26)  |           |           |           |           |           |           |  |
| fsFromStart        | 861 (17)  |           |           |           |           |           |           |           |  |
| fSharePrintInfo    | 1096*(13) | 36-(17)   | 114 (17)  | 1495 (17) | 304 (37)  | 332 (37)  | 571 (37)  | 1806 (37) |  |
| fShiftKey          | 641*( 9)  | 2789-(10) | 2837 (10) | 2990 (10) | 3013-(10) |           |           |           |  |
| fShowBorders       | 478-(16)  |           |           |           |           |           |           |           |  |
| fShowBreaks        | 160*(36)  | 267-(37)  | 679-(37)  | 679 (37)  | 704 (37)  | 740 (37)  | 789 (37)  | 895 (37)  |  |
|                    | 1628 (37) |           |           |           |           |           |           |           |  |
| fShowExtraFeedback | 479-(16)  |           |           |           |           |           |           |           |  |
| fShown             | 2069-( 6) | 2126-( 6) | 1469*(13) | 200-(18)  | 226-(18)  | 258 (18)  | 1023 (18) | 1374 (18) |  |
|                    | 1650 (18) | 1653-(18) | 1656-(18) | 1916 (18) | 933-(19)  |           |           |           |  |
| fSingleSelection   | 149*( 9)  | 119-(10)  | 168-(10)  | 267 (10)  | 2060-(10) | 2158 (10) | 2958 (10) | 2990 (10) |  |
|                    | 3039 (10) |           |           |           |           |           |           |           |  |
| fSize              | 2591 ( 6) | 3346 ( 6) | 3347 ( 6) | 220 (10)  | 220 (10)  | 2223 (10) | 2256 (10) | 51*(11)   |  |
|                    | 59-(12)   | 85 (12)   | 88 (12)   | 111 (12)  | 114 (12)  | 210-(12)  | 210 (12)  | 229-(12)  |  |
|                    | 265 (12)  | 277 (12)  | 302 (12)  | 322 (12)  | 410 (12)  | 432 (12)  | 434 (12)  | 464-(12)  |  |
|                    | 464 (12)  | 487 (12)  | 499 (12)  | 502 (12)  | 517 (12)  | 519 (12)  | 576 (12)  | 629 (12)  |  |
|                    | 635 (12)  | 687 (12)  | 694 (12)  | 742 (12)  | 751 (12)  | 1465*(13) | 862 (14)  | 863 (14)  |  |
|                    | 925 (14)  | 2159 (16) | 195-(18)  | 255 (18)  | 364 (18)  | 381 (18)  | 383 (18)  | 383 (18)  |  |
|                    | 560 (18)  | 572 (18)  | 636 (18)  | 641 (18)  | 698 (18)  | 729 (18)  | 1034 (18) | 1139 (18) |  |
|                    | 1149 (18) | 1150 (18) | 1183 (18) | 1568 (18) | 1568 (18) | 1570 (18) | 1570 (18) | 1572 (18) |  |
|                    | 1573 (18) | 1616 (18) | 1617 (18) | 1716 (18) | 1911 (18) | 110 (19)  | 110 (19)  | 822 (19)  |  |
|                    | 822 (19)  | 831 (19)  | 831 (19)  | 835 (19)  | 835 (19)  | 165 (20)  | 225 (20)  | 226 (20)  |  |
|                    | 237 (20)  | 238 (20)  | 568 (20)  | 614 (20)  | 654 (20)  | 234 (21)  | 235 (21)  | 264 (21)  |  |
|                    | 264 (21)  | 385 (21)  | 385 (21)  | 443 (21)  | 443 (21)  | 541 (21)  | 541 (21)  | 83 (23)   |  |
|                    | 786 (37)  | 942 (37)  | 204 (39)  | 283 (40)  | 287 (40)  | 518 (40)  | 521 (40)  | 533 (40)  |  |
|                    | 534 (40)  | 630 (40)  | 631 (40)  | 1221 (40) | 1222 (40) | 1264 (40) | 1269 (40) | 1270 (40) |  |
|                    | 1279 (40) | 1279 (40) | 1320 (40) |           |           |           |           |           |  |
| fSizeable          | 2065*(13) | 110-(21)  | 142-(21)  | 181 (21)  | 232 (21)  | 380 (21)  | 440 (21)  | 510 (21)  |  |
|                    | 789 (21)  |           |           |           |           |           |           |           |  |
| fSizeDeterminer    | 219 (10)  | 2221 (10) | 2254 (10) | 2461-(10) | 1466*(13) | 196-(18)  | 197-(18)  | 256 (18)  |  |
|                    | 257 (18)  | 632 (18)  | 1718 (18) | 1720 (18) | 1912 (18) | 1913 (18) | 286 (40)  | 373 (40)  |  |
| fSpecsChanged      | 132*(38)  | 44-(40)   | 93-(40)   | 228-(40)  | 709-(40)  | 867-(40)  | 890-(40)  | 1194-(40) |  |
|                    | 1430-(40) | 1655 (40) | 151-(41)  | 240-(41)  | 560-(41)  | 744-(41)  |           |           |  |
| fSquareDots        | 173*(36)  | 276-(37)  | 898 (37)  | 1724 (37) |           |           |           |           |  |
| fsRdPerm           | 74 (17)   | 75 (17)   |           |           |           |           |           |           |  |
| fsRdWrPerm         | 704 (17)  | 704 (17)  |           |           |           |           |           |           |  |
| FSRead             | 379 (17)  |           |           |           |           |           |           |           |  |
| fStaggered         | 1925*(13) | 49-(19)   | 197 (19)  | 951-(19)  | 1042 (19) |           |           |           |  |
| fStdPrintPage      | 133*(36)  | 265-(37)  | 890 (37)  | 1110 (37) | 1231 (37) | 1471 (37) | 2040 (37) |           |  |
| fStdPrintHandler   | 483*(36)  | 2105-(37) | 2140 (37) | 2153 (37) | 2167 (37) | 2168 (37) | 2179 (37) | 2180 (37) |  |
| fStylePad          | 396*(38)  | 154 (39)  | 48-(41)   | 81-(41)   | 113 (41)  | 178 (41)  | 195 (41)  | 495-(41)  |  |
|                    | 496 (41)  | 501 (41)  | 685 (41)  | 751-(41)  | 751 (41)  | 775 (41)  | 796 (41)  |           |  |
| fStyleType         | 128*(38)  | 40-(40)   | 103-(40)  | 141 (40)  | 310 (40)  | 596 (40)  | 657 (40)  | 1127 (40) |  |
|                    | 1224 (40) | 1371 (40) | 1426 (40) | 1427 (40) | 1493 (40) | 1541 (40) | 1605 (40) | 1652 (40) |  |

|                   |            |            |            |            |            |            |            |            |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                   | 76 (41)    | 140 (41)   | 229 (41)   | 378 (41)   | 390 (41)   | 466 (41)   | 489 (41)   | 585 (41)   |
|                   | 735 (41)   | 916 (41)   |            |            |            |            |            |            |
| fSubviews         | 1462* (13) | 2761 (16)  | 192- (18)  | 289 (18)   | 290 (18)   | 291- (18)  | 342 (18)   | 344- (18)  |
|                   | 346 (18)   | 350 (18)   | 354 (18)   | 697 (18)   | 698 (18)   | 939 (18)   | 940 (18)   | 990 (18)   |
|                   | 991 (18)   | 1392 (18)  | 1393 (18)  | 1462 (18)  | 1463 (18)  | 1465 (18)  | 1466 (18)  | 1477 (18)  |
|                   | 1478 (18)  | 1480 (18)  | 1481 (18)  | 1538 (18)  | 1539 (18)  | 1908 (18)  |            |            |
| fSuperview        | 1236 (6)   | 2584 (6)   | 2585 (6)   | 2586 (6)   | 2222 (10)  | 2223 (10)  | 1461* (13) | 2766- (16) |
|                   | 249 (18)   | 250 (18)   | 292 (18)   | 293 (18)   | 356- (18)  | 640 (18)   | 641 (18)   | 748 (18)   |
|                   | 749 (18)   | 1066 (18)  | 1067 (18)  | 1108 (18)  | 1126 (18)  | 1127 (18)  | 1161 (18)  | 1162 (18)  |
|                   | 1197 (18)  | 1199 (18)  | 1200 (18)  | 1227 (18)  | 1228 (18)  | 1421 (18)  | 1450 (18)  | 1451 (18)  |
|                   | 1544- (18) | 1600 (18)  | 1601 (18)  | 1628 (18)  | 1631 (18)  | 1823 (18)  | 1907 (18)  | 229 (20)   |
|                   | 241 (20)   | 90 (23)    |            |            |            |            |            |            |
| fSuperview        | 191- (18)  |            |            |            |            |            |            |            |
| fWrite            | 424 (17)   |            |            |            |            |            |            |            |
| fTarget           | 1909* (13) | 2437* (13) | 874- (14)  | 951- (14)  | 1037- (14) | 2285- (16) | 2770- (16) | 54- (19)   |
|                   | 130- (19)  | 273 (19)   | 278- (19)  | 895- (19)  | 1035 (19)  | 17- (22)   | 127 (22)   |            |
| fTargetID         | 1910* (13) | 1033 (14)  | 1035 (14)  | 55- (19)   | 131- (19)  | 201 (19)   | 1036 (19)  |            |
| fTEView           | 61* (5)    | 565* (5)   | 272- (6)   | 287- (6)   | 315- (6)   | 332- (6)   | 383 (6)    | 522 (6)    |
|                   | 763 (6)    | 2607- (6)  | 2931- (6)  | 2954- (6)  | 3010 (6)   | 3011 (6)   | 3024 (6)   | 3026 (6)   |
|                   | 3045 (6)   | 3047 (6)   | 3074 (6)   | 3091 (6)   | 3092 (6)   | 3119 (6)   | 3121 (6)   | 3141 (6)   |
|                   | 3158 (6)   | 3162 (6)   | 3222 (6)   | 3223 (6)   | 3235 (6)   | 3236 (6)   | 3247 (6)   | 3248 (6)   |
|                   | 3258 (6)   | 3259 (6)   | 3270 (6)   | 3271 (6)   | 3272 (6)   | 3274 (6)   | 3288 (6)   | 3293 (6)   |
|                   | 3294 (6)   | 3321 (6)   | 3322 (6)   | 3367 (6)   | 3368- (6)  | 372* (38)  | 28- (41)   | 129 (41)   |
|                   | 140 (41)   | 151 (41)   | 166 (41)   | 223 (41)   | 229 (41)   | 240 (41)   | 255 (41)   | 265 (41)   |
|                   | 281 (41)   | 287 (41)   | 301 (41)   | 363 (41)   | 373 (41)   | 544 (41)   | 554 (41)   | 557 (41)   |
|                   | 558 (41)   | 560 (41)   | 585 (41)   | 667 (41)   | 668 (41)   | 735 (41)   | 744 (41)   | 860 (41)   |
|                   | 916 (41)   |            |            |            |            |            |            |            |
| fText             | 98* (38)   | 29- (40)   | 56- (40)   | 82- (40)   | 111- (40)  | 188 (40)   | 730 (40)   | 854 (40)   |
|                   | 1052 (40)  | 1113 (40)  | 1122 (40)  | 1534- (40) | 1603 (40)  | 1640 (40)  | 129 (41)   | 223 (41)   |
| fTextPad          | 395* (38)  | 114 (39)   | 47- (41)   | 62- (41)   | 113 (41)   | 177 (41)   | 195 (41)   | 487- (41)  |
|                   | 501 (41)   | 682- (41)  | 682 (41)   | 685 (41)   | 781- (41)  | 796 (41)   | 799- (41)  | 799 (41)   |
| fTextStyle        | 1263 (6)   | 2589 (6)   | 2817 (6)   | 2936- (6)  | 418* (9)   | 2213- (10) | 2247- (10) | 2280 (10)  |
|                   | 2281 (10)  | 2286 (10)  | 2377 (10)  | 2412 (10)  | 2070* (13) | 105- (21)  | 116- (21)  | 149- (21)  |
|                   | 171 (21)   | 172 (21)   | 186 (21)   | 341 (21)   | 395 (21)   | 406- (21)  | 516 (21)   | 120* (38)  |
|                   | 36- (40)   | 100- (40)  | 134 (40)   | 149 (40)   | 1186 (40)  | 1218 (40)  | 1388 (40)  | 1420- (40) |
|                   | 1648 (40)  | 529 (41)   |            |            |            |            |            |            |
| fTheColumn        | 735* (9)   | 757* (9)   | 3116- (10) | 3130 (10)  | 3151- (10) | 3166 (10)  |            |            |
| fTheRow           | 712* (9)   | 758* (9)   | 3083- (10) | 3097 (10)  | 3150- (10) | 3165 (10)  |            |            |
| fTitle            | 1103* (13) | 298 (16)   | 2164- (16) | 33- (17)   | 165 (17)   | 248 (17)   | 331 (17)   | 506 (17)   |
|                   | 523 (17)   | 586 (17)   | 589 (17)   | 738 (17)   | 770 (17)   | 826 (17)   | 831- (17)  | 917 (17)   |
|                   | 969 (17)   | 1090 (17)  | 1228 (17)  | 1334 (17)  | 1390- (17) | 1497 (17)  | 730 (19)   | 1014 (37)  |
| fTmpHLRgn         | 154* (9)   | 117- (10)  | 217- (10)  | 296 (10)   | 501 (10)   | 502 (10)   | 1999 (10)  | 2012 (10)  |
|                   | 2017 (10)  | 2147 (10)  |            |            |            |            |            |            |
| fTmpRgn           | 153* (9)   | 111- (10)  | 211- (10)  | 294 (10)   | 552 (10)   | 553 (10)   | 872 (10)   | 873 (10)   |
|                   | 874 (10)   | 969 (10)   | 970 (10)   | 971 (10)   | 1998 (10)  | 1999 (10)  | 2006 (10)  | 2006 (10)  |
|                   | 2007 (10)  | 2146 (10)  | 2904 (10)  | 2905 (10)  | 2911 (10)  | 2914 (10)  | 2971 (10)  | 2977 (10)  |
|                   | 2983 (10)  |            |            |            |            |            |            |            |
| fTmpSelRgn        | 152* (9)   | 114- (10)  | 214- (10)  | 295 (10)   | 1937 (10)  | 1939 (10)  | 1942 (10)  | 1954 (10)  |
|                   | 1955 (10)  | 2047 (10)  | 2049 (10)  | 2145 (10)  |            |            |            |            |
| fTrackNonMovement | 2456* (13) | 3289 (16)  | 46- (21)   | 24- (22)   |            |            |            |            |
| fTranslation      | 1758* (13) | 3138 (16)  | 3163 (16)  | 3164 (16)  | 26- (20)   | 59- (20)   | 196 (20)   | 199 (20)   |
|                   | 263 (20)   | 266 (20)   | 278 (20)   | 281 (20)   | 300 (20)   | 321 (20)   | 332 (20)   | 332 (20)   |
|                   | 366 (20)   | 542 (20)   | 595 (20)   | 595 (20)   | 611 (20)   | 621 (20)   | 626 (20)   | 626 (20)   |
|                   | 664 (20)   |            |            |            |            |            |            |            |
| fType             | 356 (16)   | 458- (16)  | 458 (16)   |            |            |            |            |            |
| fTypingCommand    | 116* (38)  | 35- (40)   | 91- (40)   | 736 (40)   | 738 (40)   | 743- (40)  | 747 (40)   | 903 (40)   |
|                   | 904 (40)   | 1646 (40)  | 667 (41)   | 668- (41)  |            |            |            |            |
| fUsesDataFork     | 1117* (13) | 47- (17)   | 54 (17)    | 58- (17)   | 654 (17)   | 703 (17)   | 839 (17)   | 1505 (17)  |
| fUsesRsrcFork     | 1119* (13) | 48- (17)   | 61 (17)    | 65- (17)   | 356 (17)   | 654 (17)   | 671 (17)   | 703 (17)   |
|                   | 840 (17)   | 1506 (17)  |            |            |            |            |            |            |
| fValue            | 31* (1)    | 34- (2)    | 35 (2)     | 47 (2)     | 61 (2)     | 170 (2)    | 203- (2)   | 220 (2)    |
|                   | 247 (2)    | 306 (2)    | 346 (2)    | 682 (6)    | 682 (6)    | 1161- (14) | 75 (15)    |            |
| fVertCentered     | 1924* (13) | 48- (19)   | 199 (19)   | 354- (19)  | 1041 (19)  |            |            |            |
| fView             | 2326* (13) | 2450* (13) | 3179 (16)  | 3238 (16)  | 442 (17)   | 443 (17)   | 1016 (18)  | 25- (22)   |
|                   | 87 (22)    | 87 (22)    | 132 (22)   | 15- (24)   | 43 (24)    | 128 (24)   | 130 (24)   | 292 (37)   |
|                   | 321 (37)   | 324 (37)   | 325 (37)   | 579 (37)   | 697 (37)   | 735 (37)   | 753 (37)   | 785 (37)   |
|                   | 786 (37)   | 787 (37)   | 831 (37)   | 861 (37)   | 942 (37)   | 960 (37)   | 997 (37)   | 1019 (37)  |
|                   | 1084 (37)  | 1265 (37)  | 1468 (37)  | 1561 (37)  | 1628 (37)  | 1641 (37)  | 1653 (37)  | 1663 (37)  |
|                   | 1665 (37)  | 1776 (37)  | 1803 (37)  | 1860 (37)  | 2020 (37)  | 2021 (37)  |            |            |
|                   | 2454* (13) | 3206 (16)  | 47- (21)   | 23- (22)   | 135 (22)   |            |            |            |
| fViewConstrain    | 188* (36)  | 902 (37)   | 960 (37)   | 1861- (37) |            |            |            |            |
| fViewEnabled      | 1470* (13) | 201- (18)  | 227- (18)  | 259 (18)   | 1383 (18)  | 1784- (18) | 1917 (18)  |            |
| fViewList         | 1081* (13) | 26- (17)   | 88- (17)   | 90 (17)    | 111 (17)   | 112 (17)   | 127 (17)   | 224 (17)   |
|                   | 446 (17)   | 1000 (17)  | 1491 (17)  |            |            |            |            |            |
| fViewPerPage      | 2334* (13) | 644 (18)   | 653 (18)   | 46 (24)    | 417 (37)   | 461 (37)   | 980 (37)   | 1258 (37)  |
|                   | 1635 (37)  | 1642- (37) | 1891 (37)  | 1900 (37)  | 588 (40)   |            |            |            |
| fVolRefNum        | 1106* (13) | 38- (17)   | 1334 (17)  | 1500 (17)  |            |            |            |            |
| fVolRefnum        | 293 (16)   | 248 (17)   | 527 (17)   | 827 (17)   | 832- (17)  | 1091 (17)  | 1230- (17) |            |
| fWholeRect        | 147* (9)   | 120- (10)  | 169- (10)  | 552 (10)   | 1941- (10) | 1956- (10) | 1995 (10)  | 2028- (10) |
|                   | 2033 (10)  | 2048- (10) | 2061- (10) | 2160 (10)  | 2908 (10)  | 2987- (10) | 2994- (10) | 3000 (10)  |
|                   | 3012- (10) |            |            |            |            |            |            |            |
| fWindowList       | 1079* (13) | 2159 (16)  | 2161 (16)  | 25- (17)   | 83- (17)   | 85 (17)    | 108 (17)   | 109 (17)   |
|                   | 148 (17)   | 234 (17)   | 456 (17)   | 493 (17)   | 744 (17)   | 1490 (17)  |            |            |
| fWmgrWindow       | 1905* (13) | 1030 (14)  | 1031 (14)  | 743 (16)   | 850 (16)   | 1111 (18)  | 1113 (18)  | 1755 (18)  |
|                   | 1757 (18)  | 125- (19)  | 325 (19)   | 355 (19)   | 366 (19)   | 447 (19)   | 512 (19)   | 517 (19)   |
|                   | 616 (19)   | 617 (19)   | 632 (19)   | 652 (19)   | 652 (19)   | 653 (19)   | 665 (19)   | 697 (19)   |
|                   | 750 (19)   | 751 (19)   | 764 (19)   | 765 (19)   | 776 (19)   | 791 (19)   | 804 (19)   | 824 (19)   |
|                   | 856 (19)   | 880 (19)   | 928 (19)   | 930 (19)   | 932 (19)   | 991 (19)   | 1031 (19)  | 273 (20)   |
|                   | 877 (14)   | 954 (14)   | 2163 (16)  | 34- (19)   | 104- (19)  | 180 (19)   | 234 (19)   | 475 (19)   |
|                   | 489 (19)   | 519 (19)   | 635 (19)   | 636 (19)   | 913 (19)   | 916 (19)   | 1011 (19)  | 1013 (19)  |
|                   | 1021 (37)  |            |            |            |            |            |            |            |
| -G-               |            |            |            |            |            |            |            |            |
| gApplMemList      | 171* (27)  | 507- (28)  | 977 (28)   | 978 (28)   |            |            |            |            |
| gApp2MemList      | 172* (27)  | 508- (28)  | 980 (28)   | 981 (28)   |            |            |            |            |
| gAppDone          | 2592* (13) | 77- (16)   | 715- (16)  | 725- (16)  | 912 (16)   | 1914 (16)  | 764 (17)   |            |
| gApplication      | 655 (6)    | 2594* (13) | 95 (14)    | 96 (14)    | 454- (14)  | 996 (14)   | 28 (15)    | 76- (16)   |
|                   | 158 (16)   | 913 (16)   | 1081 (16)  | 77 (17)    | 104 (17)   | 210 (17)   | 926 (17)   | 938 (17)   |
|                   | 989 (17)   | 1108 (17)  | 239 (19)   | 273 (19)   | 279 (19)   | 279 (19)   | 702 (19)   | 703 (19)   |

|                         |            |            |            |            |            |            |            |            |
|-------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                         | 728 (19)   | 737 (19)   | 738 (19)   | 776 (19)   | 897 (19)   | 74 (37)    | 97 (37)    | 1306 (37)  |
|                         | 846 (40)   | 397 (41)   | 474 (41)   |            |            |            |            |            |
| gAskAboutAlloc          | 168* (7)   | 205* (8)   | 497 (14)   | 73 (33)    | 172 (33)   |            |            |            |
| gAskFailure             | 170* (7)   | 44 (8)     | 95 (8)     | 123 (8)    | 206* (8)   | 501 (14)   |            |            |
| gBoolString             | 538* (14)  | 539* (14)  | 199* (25)  | 940 (26)   | 250 (30)   | 275 (30)   | 275 (30)   |            |
| gBreaksPenState         | 514* (36)  | 184 (37)   | 734 (37)   | 752 (37)   | 786 (37)   |            |            |            |
| gBusyTempRgn            | 2596* (13) | 129 (14)   | 131* (14)  | 593* (14)  | 1272 (14)  | 1280* (14) | 914 (16)   |            |
| gCancelAllPrinting      | 516* (36)  | 84* (37)   | 178* (37)  | 1544* (37) | 1551 (37)  |            |            |            |
| gChooserOK              | 2598* (13) | 657* (14)  | 915 (16)   | 194 (37)   |            |            |            |            |
| gClickCount             | 2610* (13) | 571* (14)  | 590 (16)   | 593 (16)   | 599* (16)  | 916 (16)   | 1405 (16)  |            |
| gClipClaimed            | 2619* (13) | 492* (16)  | 917 (16)   | 2255 (16)  | 2291* (16) |            |            |            |
| gClipOrphanage          | 2622* (13) | 146 (16)   | 147 (16)   | 148 (16)   | 397 (16)   | 918 (16)   | 1941 (16)  | 2498 (16)  |
|                         | 2771 (16)  | 287 (23)   |            |            |            |            |            |            |
| gClipUndoView           | 2626* (13) | 664* (14)  | 162 (16)   | 175 (16)   | 178 (16)   | 181 (16)   | 182* (16)  | 471* (16)  |
|                         | 919 (16)   | 2257 (16)  | 2258* (16) | 2982 (16)  | 2983* (16) |            |            |            |
| gClipView               | 2629* (13) | 75 (14)    | 76 (14)    | 663* (14)  | 161 (16)   | 178 (16)   | 222 (16)   | 227 (16)   |
|                         | 386 (16)   | 387* (16)  | 471 (16)   | 920 (16)   | 1100 (16)  | 2780* (16) | 2983 (16)  | 285 (23)   |
|                         | 917 (40)   | 491 (41)   |            |            |            |            |            |            |
| gClipWindow             | 2631* (13) | 659* (14)  | 150* (16)  | 743 (16)   | 744 (16)   | 747 (16)   | 748 (16)   | 850 (16)   |
|                         | 921 (16)   | 2758 (16)  | 2760 (16)  | 2761 (16)  | 2763 (16)  | 2769 (16)  | 2770 (16)  | 2770 (16)  |
|                         | 195 (17)   | 493 (17)   |            |            |            |            |            |            |
| gClipWrittenToDeskScrap | 2633* (13) | 223 (16)   | 229* (16)  | 922 (16)   | 2771* (16) | 272* (23)  |            |            |
| gCmdTable               | 25* (30)   | 70 (30)    | 119 (30)   | 323* (30)  | 324 (30)   | 325 (30)   | 331 (30)   | 336 (30)   |
| gCodeReserve            | 28* (28)   | 244 (28)   | 276 (28)   | 421* (28)  | 629 (28)   | 631 (28)   | 645 (28)   | 649 (28)   |
|                         | 661 (28)   | 663 (28)   | 672 (28)   | 1158 (28)  |            |            |            |            |
| gCodeSegs               | 494 (14)   | 190* (27)  | 444* (28)  | 452* (28)  | 459 (28)   | 461 (28)   | 776 (28)   | 976 (28)   |
|                         | 1024 (28)  |            |            |            |            |            |            |            |
| gConfiguration          | 77 (4)     | 127 (4)    | 145 (4)    | 175 (4)    | 208 (4)    | 78 (6)     | 116 (6)    | 167 (6)    |
|                         | 1412 (6)   | 1458 (6)   | 1635 (6)   | 1681 (6)   | 2044 (6)   | 2098 (6)   | 2267 (6)   | 2652 (6)   |
|                         | 3188 (6)   | 3192 (6)   | 399 (14)   | 404 (14)   | 422 (14)   | 583 (14)   | 609 (14)   | 613 (14)   |
|                         | 678 (14)   | 682 (14)   | 700 (14)   | 701 (14)   | 702 (14)   | 703 (14)   | 1077 (14)  | 24 (16)    |
|                         | 27 (16)    | 131 (16)   | 1137 (16)  | 1264 (16)  | 1275 (16)  | 532 (17)   | 688 (17)   | 1842 (18)  |
|                         | 1891 (18)  | 119 (19)   | 593 (19)   | 197* (25)  | 63 (26)    | 64 (26)    | 208 (26)   | 278 (26)   |
|                         | 299 (26)   | 364 (26)   | 365 (26)   | 373 (26)   | 612 (26)   | 635 (26)   | 647 (26)   | 768 (26)   |
|                         | 787 (26)   | 1152 (26)  | 514 (28)   | 218 (30)   | 371 (30)   | 453 (30)   | 76 (35)    | 105 (35)   |
|                         | 41 (40)    | 103 (40)   | 204 (40)   | 249 (40)   | 433 (40)   | 490 (40)   | 1185 (40)  | 1337 (40)  |
|                         | 1392 (40)  | 533 (41)   | 719 (41)   | 793 (41)   |            |            |            |            |
| gCopyright              | 47* (14)   | 535* (14)  |            |            |            |            |            |            |
| gCouldPrint             | 2635* (13) | 637* (14)  | 923 (16)   | 192* (37)  | 637 (37)   | 697 (37)   | 1720 (37)  | 1737 (37)  |
| gCurrPrintHandler       | 2637* (13) | 633* (14)  | 1139 (14)  | 924 (16)   | 1016 (18)  | 1016 (18)  | 1018 (18)  | 95 (37)    |
|                         | 96* (37)   | 99* (37)   | 1158* (37) | 1181* (37) |            |            |            |            |
| gCurrTEView             | 9* (39)    | 191 (39)   | 193 (39)   | 197 (39)   | 199 (39)   | 203 (39)   | 204 (39)   | 204 (39)   |
|                         | 223 (39)   | 224 (39)   | 224 (39)   | 888* (40)  | 1188* (40) |            |            |            |
| gCursorInfo             | 37* (4)    | 73 (4)     | 74 (4)     | 91 (4)     | 116 (4)    | 117 (4)    | 128 (4)    | 183 (4)    |
|                         | 204 (4)    | 235 (4)    | 240 (4)    | 274 (4)    | 294 (4)    | 298 (4)    | 322 (4)    | 324 (4)    |
|                         | 325 (4)    | 328 (4)    |            |            |            |            |            |            |
| gCursorRgn              | 2639* (13) | 579* (14)  | 805 (14)   | 925 (16)   | 2355 (16)  | 2794 (16)  | 3018 (16)  | 3038 (16)  |
|                         | 3046 (16)  | 3051 (16)  | 3059 (16)  | 3059 (16)  | 3065 (16)  | 3074 (16)  | 3078 (16)  | 3082 (16)  |
|                         | 3083 (16)  | 3085 (16)  |            |            |            |            |            |            |
| gDebugKeyMap            | 483* (14)  | 549 (14)   | 550 (14)   | 550 (14)   |            |            |            |            |
| gDebugPrinting          | 224* (13)  | 2641* (13) | 429* (14)  | 504 (14)   | 768* (16)  | 768 (16)   | 926 (16)   | 2892 (16)  |
|                         | 122 (37)   | 359 (37)   | 740 (37)   | 745 (37)   | 796 (37)   | 1139 (37)  | 1628 (37)  | 1647 (37)  |
|                         | 1656 (37)  | 1864 (37)  | 649 (40)   | 662 (40)   |            |            |            |            |
| gDebugWindowPtr         | 394* (14)  | 757 (16)   | 758 (16)   | 856 (16)   | 1311 (16)  | 1410 (16)  | 1569 (16)  | 1569 (16)  |
|                         | 1583 (16)  | 1585 (16)  | 1586 (16)  | 1682 (16)  | 1712 (16)  | 3380 (16)  | 15 (26)    |            |
| gdh                     | 149* (6)   | 171* (6)   | 173 (6)    |            |            |            |            |            |
| GDHandle                | 149 (6)    |            |            |            |            |            |            |            |
| gDocList                | 2644* (13) | 90* (16)   | 92 (16)    | 159 (16)   | 254 (16)   | 609 (16)   | 928 (16)   | 1053 (16)  |
| gDocument               | 2643* (13) | 97* (16)   | 157 (16)   | 610 (16)   | 611* (16)  | 927 (16)   | 2885 (16)  | 274* (19)  |
|                         | 281* (19)  |            |            |            |            |            |            |            |
| gdPMap                  | 173 (6)    |            |            |            |            |            |            |            |
| gDrawingPictScrap       | 2645* (13) | 634* (14)  | 1142 (14)  | 929 (16)   | 158 (18)   | 907 (18)   | 1853* (18) | 1858* (18) |
|                         | 988 (40)   |            |            |            |            |            |            |            |
| gEAPatch                | 38* (4)    | 155 (4)    | 193 (4)    |            |            |            |            |            |
| genericAlert            | 171* (14)  | 182* (14)  | 194* (14)  | 206 (14)   | 225 (14)   |            |            |            |
| gErrorParm3             | 2651* (13) | 217 (14)   | 670* (14)  | 930 (16)   | 1451* (16) | 2044* (16) | 2046 (16)  | 2047 (16)  |
|                         | 2200* (16) | 2365* (16) | 969* (17)  | 1069* (17) | 1538 (37)  |            |            |            |
| GetAppFiles             | 1496 (16)  |            |            |            |            |            |            |            |
| GetAppFont              | 373 (26)   |            |            |            |            |            |            |            |
| GetAppLimit             | 1048 (28)  |            |            |            |            |            |            |            |
| GetAppParms             | 587 (17)   |            |            |            |            |            |            |            |
| GetBackColor            | 79 (6)     |            |            |            |            |            |            |            |
| GetBreakCoord           | 239* (36)  | 803 (37)   | 971* (37)  | 1855 (37)  | 1856 (37)  |            |            |            |
| GetCaretTime            | 58 (40)    | 113 (40)   | 683 (40)   |            |            |            |            |            |
| GetCIcon                | 1413 (6)   | 1459 (6)   |            |            |            |            |            |            |
| GetClip                 | 292 (14)   | 925 (18)   | 486 (19)   | 1000 (40)  | 1016 (40)  |            |            |            |
| GetColWidth             | 315* (9)   | 260 (10)   | 467 (10)   | 733 (10)   | 782 (10)   | 801 (10)   | 1342* (10) | 1360* (10) |
|                         | 1362* (10) | 1364* (10) | 2222 (10)  | 2254 (10)  | 2321 (10)  | 2602 (10)  |            |            |
| GetCRefCon              | 20 (21)    |            |            |            |            |            |            |            |
| GetCtlMax               | 724 (21)   |            |            |            |            |            |            |            |
| GetCtlMin               | 734 (21)   |            |            |            |            |            |            |            |
| GetCtlValue             | 753 (21)   |            |            |            |            |            |            |            |
| GetCurrentItem          | 478* (5)   | 2419* (6)  | 2421* (6)  |            |            |            |            |            |
| GetCursor               | 116 (4)    | 920 (40)   |            |            |            |            |            |            |
| GetDataToPaste          | 1027* (13) | 1090* (16) | 1112* (16) | 846 (40)   | 474 (41)   |            |            |            |
| GetDateTime             | 595 (17)   |            |            |            |            |            |            |            |
| GetDblTime              | 591 (16)   |            |            |            |            |            |            |            |
| GetDefFontSize          | 679 (14)   | 64 (26)    | 250 (40)   |            |            |            |            |            |
| GetDialogView           | 102* (5)   | 561* (6)   | 563* (6)   | 1280 (6)   | 2792 (6)   | 1637* (13) | 1124* (18) | 1127* (18) |
|                         | 1127 (18)  | 1129* (18) |            |            |            |            |            |            |
| GetDirID                | 294 (16)   | 308 (16)   | 402* (25)  | 197 (26)   | 204* (26)  | 218* (26)  | 225* (26)  |            |
| GetDItem                | 1398 (37)  |            |            |            |            |            |            |            |
| getDlgID                | 2933 (16)  |            |            |            |            |            |            |            |
| GetDocName              | 261* (36)  | 1009* (37) | 1397 (37)  | 1538 (37)  | 2000 (37)  |            |            |            |
| GetDriverName           | 154* (37)  | 526 (37)   |            |            |            |            |            |            |
| GetEnvirons             | 754 (26)   | 495 (40)   |            |            |            |            |            |            |
| GetEvent                | 703* (13)  | 1119* (16) | 1200* (16) | 2355 (16)  | 3367 (16)  | 702 (19)   |            |            |
| GetExtent               | 3344 (6)   | 1645* (13) | 1805* (13) | 3133 (16)  | 3182 (16)  | 687 (18)   | 1081 (18)  | 1136* (18) |
|                         | 1173 (18)  | 1216 (18)  | 193 (20)   | 328* (20)  | 331 (20)   | 439 (20)   | 489 (20)   |            |

|                  |            |            |            |            |            |            |            |            |
|------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| GetFileInfo      | 360 (16)   | 248 (17)   | 1144 (17)  | 1181 (17)  | 406* (25)  | 176 (26)   | 233* (26)  | 246- (26)  |
| GetFNum          | 265 (26)   |            |            |            |            |            |            |            |
| GetFocus         | 286* (14)  | 3130 (16)  | 3154 (16)  | 909 (21)   | 1058 (21)  | 1361 (21)  |            |            |
| GetFontInfo      | 1264 (6)   | 2357 (10)  | 323 (40)   | 447 (40)   | 1408 (40)  | 1559 (40)  |            |            |
| GetFontName      | 381 (26)   | 942 (26)   | 134 (40)   |            |            |            |            |            |
| GetFontNum       | 2244 (10)  | 148 (21)   | 252* (25)  | 253* (26)  | 266- (26)  | 99 (40)    |            |            |
| GetForeColor     | 300 (26)   |            |            |            |            |            |            |            |
| GetFrame         | 1649* (13) | 895 (18)   | 1028 (18)  | 1146* (18) | 1571 (18)  | 1584 (18)  |            |            |
| GetGlobalBounds  | 2018* (13) | 3043 (16)  | 319 (19)   | 359 (19)   | 562 (19)   | 626* (19)  | 953 (19)   |            |
| GetGrafPort      | 1639* (13) | 2016* (13) | 360 (18)   | 1157* (18) | 1160- (18) | 1162- (18) | 1162 (18)  | 1164- (18) |
|                  | 662* (19)  | 665- (19)  | 637 (21)   | 1628 (37)  | 1240 (40)  |            |            |            |
| GetGrayRgn       | 3152 (16)  |            |            |            |            |            |            |            |
| GetHandleBits    | 1547 (6)   | 1767 (6)   | 1965 (6)   | 258* (25)  | 274* (26)  | 279- (26)  | 282- (26)  | 921 (28)   |
|                  | 1076 (28)  |            |            |            |            |            |            |            |
| GetHandleSize    | 679 (6)    | 685 (6)    | 3163 (6)   | 1559 (10)  | 1573 (10)  | 1587 (10)  | 1659 (10)  | 1673 (10)  |
|                  | 1687 (10)  | 62 (12)    | 450 (12)   | 461 (12)   | 542 (12)   | 742 (14)   | 1122 (14)  | 30 (16)    |
|                  | 351 (16)   | 428 (16)   | 107 (18)   | 245 (23)   | 649 (28)   | 663 (28)   | 677 (28)   | 964 (28)   |
|                  | 1079 (28)  | 325 (30)   | 112 (32)   | 67 (39)    | 115 (39)   | 133 (39)   | 150 (39)   | 186 (40)   |
|                  | 730 (40)   | 854 (40)   | 1113 (40)  | 1143 (40)  | 1537 (40)  | 81 (41)    | 130 (41)   | 142 (41)   |
|                  | 144 (41)   | 148 (41)   | 224 (41)   | 237 (41)   | 747 (41)   | 775 (41)   | 866 (41)   | 868 (41)   |
| GetIcon          | 1417 (6)   | 1463 (6)   |            |            |            |            |            |            |
| GetIfBkColor     | 69* (6)    | 2262 (6)   | 2311 (6)   |            |            |            |            |            |
| GetIfColor       | 1262 (6)   | 2262 (6)   | 2311 (6)   | 2816 (6)   | 261* (25)  | 290* (26)  |            |            |
| GetIndResource   | 115 (28)   | 486 (28)   | 493 (28)   | 809 (28)   |            |            |            |            |
| GetIndString     | 1142 (6)   | 2692 (6)   | 202 (14)   | 758 (14)   | 2818 (16)  | 166 (17)   | 769 (17)   | 1417 (17)  |
|                  | 1468 (17)  | 565 (30)   |            |            |            |            |            |            |
| GetInspectorName | 246* (11)  | 651* (12)  | 695* (13)  | 1157* (13) | 1735* (13) | 2035* (13) | 2344* (13) | 2578* (13) |
|                  | 1078* (16) | 503* (17)  | 1103* (18) | 649* (19)  | 282* (23)  | 120* (24)  | 94* (31)   | 264* (33)  |
| GetItem          | 2376 (6)   | 2431 (6)   | 1998 (16)  | 159 (30)   |            |            |            |            |
| GetItemCmd       | 455 (30)   |            |            |            |            |            |            |            |
| GetItemHeight    | 582* (9)   | 2589* (10) | 2591- (10) |            |            |            |            |            |
| GetItemNumber    | 91* (11)   | 260* (11)  | 378* (12)  | 399- (12)  | 677* (12)  | 690 (12)   | 707- (12)  | 718 (12)   |
|                  | 350 (18)   |            |            |            |            |            |            |            |
| GetItemText      | 480* (5)   | 2428* (6)  | 552* (9)   | 2611* (10) | 2629 (10)  |            |            |            |
| GetItemWidth     | 585* (9)   | 2600* (10) | 2602- (10) |            |            |            |            |            |
| GetIText         | 1401 (37)  |            |            |            |            |            |            |            |
| GetKeys          | 549 (14)   |            |            |            |            |            |            |            |
| GetLabel         | 275* (5)   | 1187 (6)   | 1301* (6)  |            |            |            |            |            |
| GetLongMax       | 2251* (13) | 1081* (21) | 1083- (21) |            |            |            |            |            |
| GetLongMin       | 2253* (13) | 1090* (21) | 1092- (21) |            |            |            |            |            |
| GetLongVal       | 2255* (13) | 1099* (21) | 1101- (21) |            |            |            |            |            |
| GetMax           | 2180* (13) | 721* (21)  | 724- (21)  | 1067 (21)  | 1365 (21)  |            |            |            |
| GetMaxDevice     | 171 (6)    |            |            |            |            |            |            |            |
| GetMBarHeight    | 614 (14)   |            |            |            |            |            |            |            |
| GetMCEntry       | 175 (6)    | 184 (6)    | 211 (6)    |            |            |            |            |            |
| GetMCInfo        | 25 (16)    |            |            |            |            |            |            |            |
| GetMenu          | 2051 (6)   | 2107 (6)   | 119 (16)   | 138 (16)   |            |            |            |            |
| GetMenuColors    | 145* (6)   | 190 (6)    | 2263 (6)   | 2313 (6)   | 2318 (6)   |            |            |            |
| GetMenuList      | 43* (30)   | 199 (30)   |            |            |            |            |            |            |
| GetMethodName    | 47 (8)     | 98 (8)     | 126 (8)    | 430* (25)  | 331* (26)  | 76 (33)    |            |            |
| GetMHandle       | 1998 (16)  | 2858 (16)  | 253 (30)   | 278 (30)   |            |            |            |            |
| GetMin           | 2182* (13) | 731* (21)  | 734- (21)  |            |            |            |            |            |
| GetMouse         | 3015 (16)  | 3274 (16)  | 189 (39)   |            |            |            |            |            |
| GetNamedResource | 2086 (16)  | 177 (28)   | 819 (28)   | 51 (32)    |            |            |            |            |
| GetNewCWindow    | 1276 (16)  |            |            |            |            |            |            |            |
| GetNewDialog     | 1392 (37)  | 1997 (37)  |            |            |            |            |            |            |
| GetNewMBar       | 26 (16)    |            |            |            |            |            |            |            |
| GetNewWindow     | 1278 (16)  |            |            |            |            |            |            |            |
| GetNextEvent     | 1186 (16)  | 67 (37)    |            |            |            |            |            |            |
| GetPattern       | 1640 (6)   | 1686 (6)   |            |            |            |            |            |            |
| GetPenState      | 479 (18)   |            |            |            |            |            |            |            |
| GetPicture       | 1855 (6)   | 1891 (6)   |            |            |            |            |            |            |
| GetPixPat        | 1636 (6)   | 1682 (6)   |            |            |            |            |            |            |
| GetPort          | 289 (14)   | 634 (19)   | 66 (37)    | 1600 (37)  | 320 (40)   | 444 (40)   | 991 (40)   | 1216 (40)  |
|                  | 1405 (40)  |            |            |            |            |            |            |            |
| GetPortFontInfo  | 2281 (10)  | 172 (21)   | 264* (25)  | 357* (26)  |            |            |            |            |
| GetPrintExtent   | 1720* (13) | 1171* (18) | 1776 (37)  | 347* (38)  | 1059* (40) | 1062 (40)  |            |            |
| GetProcName      | 434* (25)  | 339 (26)   | 389* (26)  | 176 (33)   | 179 (33)   |            |            |            |
| GetProcName      | 159 (8)    |            |            |            |            |            |            |            |
| GetQDExtent      | 3122 (6)   | 1647* (13) | 1180* (18) | 1840 (18)  | 324 (21)   | 712 (21)   | 90 (23)    | 243 (23)   |
|                  | 253 (23)   | 921 (40)   |            |            |            |            |            |            |
| GetReserveSize   | 2212 (16)  | 316* (27)  | 550* (28)  |            |            |            |            |            |
| GetResInfo       | 2097 (16)  | 116 (28)   | 776 (28)   | 822 (28)   |            |            |            |            |
| GetResLoad       | 271* (25)  | 110 (28)   | 161 (28)   | 447 (28)   | 602 (28)   |            |            |            |
| GetResMenu       | 2268 (6)   | 124 (16)   | 221* (29)  | 157 (30)   | 295* (30)  | 297- (30)  | 527 (30)   | 549 (30)   |
|                  | 606 (30)   |            |            |            |            |            |            |            |
| GetResource      | 439 (14)   | 514 (14)   | 739 (14)   | 129 (15)   | 38 (16)    | 133 (16)   | 1254 (16)  | 121 (28)   |
|                  | 451 (28)   | 604 (28)   | 913 (28)   | 297 (30)   | 323 (30)   | 373 (30)   |            |            |
| GetRowHeight     | 316* (9)   | 256 (10)   | 752 (10)   | 784 (10)   | 792 (10)   | 1373* (10) | 1391- (10) | 1393- (10) |
|                  | 1395- (10) | 1821 (10)  | 2591 (10)  |            |            |            |            |            |
| GetRsrcWindow    | 921* (13)  | 996 (14)   | 1210* (16) | 1296- (16) |            |            |            |            |
| GetSaveInfo      | 1248* (13) | 515* (17)  | 553- (17)  | 1257 (17)  | 1345 (17)  |            |            |            |
| GetScrap         | 674 (18)   | 1248 (18)  | 180 (23)   | 211 (23)   |            |            |            |            |
| GetScript        | 495 (40)   |            |            |            |            |            |            |            |
| GetScroller      | 2785 (10)  | 1641* (13) | 1850* (13) | 1190* (18) | 1196- (18) | 1199 (18)  | 1201- (18) | 339* (20)  |
|                  | 342- (20)  |            |            |            |            |            |            |            |
| GetSegNumber     | 357 (14)   | 2696 (16)  | 221* (27)  | 565* (28)  | 578- (28)  | 580- (28)  | 586- (28)  |            |
| GetSegSize       | 224* (27)  | 597* (28)  | 604- (28)  |            |            |            |            |            |
| GetSize          | 78* (11)   | 406* (12)  | 410- (12)  |            |            |            |            |            |
| gETSPatch        | 48* (14)   | 93 (14)    | 456 (14)   |            |            |            |            |            |
| GetString        | 158 (37)   |            |            |            |            |            |            |            |
| GetStylHandle    | 609 (40)   | 1040 (40)  | 1495 (40)  | 1543 (40)  |            |            |            |            |
| GetStylScrap     | 1138 (40)  | 1608 (40)  | 79 (41)    |            |            |            |            |            |
| GetSysFont       | 365 (26)   |            |            |            |            |            |            |            |
| GetSysJust       | 3189 (6)   |            |            |            |            |            |            |            |
| GetTempName      | 1257* (13) | 571* (17)  | 1347 (17)  |            |            |            |            |            |
| GetText          | 534* (5)   | 597* (5)   | 828 (6)    | 910 (6)    | 1037 (6)   | 2597 (6)   | 2740 (6)   | 2833* (6)  |
|                  | 3151* (6)  | 3159 (6)   | 3290 (6)   | 3366 (6)   | 3483 (6)   | 3519 (6)   | 469* (9)   | 588* (9)   |
|                  | 2319 (10)  | 2389* (10) | 2627* (10) | 2184* (13) | 741* (21)  |            |            |            |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| GetTrapAddress    | 103 (26)  |           |           |           |           |           |           |           |  |
| GetTrapType       | 275*(25)  | 104 (26)  | 454*(26)  | 459=(26)  | 461=(26)  | 84 (35)   | 121 (35)  | 131 (35)  |  |
|                   | 165 (35)  | 200 (35)  | 244 (35)  |           |           |           |           |           |  |
| GetVal            | 949 ( 6)  | 1078 ( 6) | 2186*(13) | 750*(21)  | 753=(21)  | 1067 (21) | 1070 (21) | 1365 (21) |  |
|                   | 1368 (21) |           |           |           |           |           |           |           |  |
| GetValue          | 658*( 5)  | 3454 ( 6) | 3476*( 6) | 3485=( 6) |           |           |           |           |  |
| GetVisibleRect    | 1651*(13) | 3054 (16) | 906 (18)  | 1209*(18) | 199 (39)  | 1306 (40) |           |           |  |
| GetVol            | 673 (17)  | 637 (26)  |           |           |           |           |           |           |  |
| GetWindow         | 525 ( 6)  | 649 ( 6)  | 3340 ( 6) | 1643*(13) | 2020*(13) | 1110 (18) | 1225*(18) | 1228=(18) |  |
|                   | 1228 (18) | 1230=(18) | 1752 (18) | 673*(19)  | 676=(19)  | 272 (20)  | 1019 (37) | 2021 (37) |  |
| GetWmgrPort       | 3150 (16) |           |           |           |           |           |           |           |  |
| GetWRefCon        | 3383 (16) | 43 (19)   |           |           |           |           |           |           |  |
| GetWTitle         | 2163 (16) | 1113 (18) | 65 (19)   | 141 (19)  | 180 (19)  | 653 (19)  | 913 (19)  | 1021 (37) |  |
| gEventLevel       | 2660*(13) | 643=(14)  | 847 (16)  | 931 (16)  | 2288 (16) | 2336 (16) | 2346=(16) | 2346 (16) |  |
|                   | 2383=(16) | 2383 (16) | 2959 (16) |           |           |           |           |           |  |
| gExperimenting    | 221*(13)  | 2663*(13) | 428=(14)  | 508 (14)  | 764=(16)  | 764 (16)  | 932 (16)  | 1473 (16) |  |
|                   | 2890 (16) |           |           |           |           |           |           |           |  |
| gFakeWindow       | 2666*(13) | 582 (14)  | 587 (14)  |           |           |           |           |           |  |
| GFIELDTOSTRTN     | 202*(25)  |           |           |           |           |           |           |           |  |
| gfieldToStrRtn    | 396=(14)  |           |           |           |           |           |           |           |  |
| gFileCount        | 2670*(13) | 639 (14)  | 933 (16)  | 1474 (16) | 1479 (16) | 1492 (16) |           |           |  |
| gFinderHPrint     | 523*(36)  | 180=(37)  | 1941 (37) | 1945 (37) | 1947=(37) | 1948 (37) | 1950 (37) | 1954 (37) |  |
| gFinderPrinting   | 2671*(13) | 636=(14)  | 640=(14)  | 102 (16)  | 934 (16)  | 1487 (16) | 1502 (16) | 2692 (16) |  |
|                   | 2697 (16) | 2702 (16) | 2706 (16) | 439 (17)  | 1382 (37) |           |           |           |  |
| gFocusedView      | 2673*(13) | 293 (14)  | 671=(14)  | 1198=(14) | 165 (16)  | 935 (16)  | 3153=(16) | 893=(18)  |  |
|                   | 1013 (18) | 1019=(18) | 1049=(18) | 1056=(18) | 1446=(18) | 1574=(18) | 259=(19)  | 509 (19)  |  |
|                   | 523=(19)  | 280 (20)  | 1369=(40) |           |           |           |           |           |  |
| gFreeWindowList   | 2674*(13) | 651=(14)  | 654 (14)  | 87 (16)   | 160 (16)  | 263 (16)  | 620 (16)  | 874 (16)  |  |
|                   | 936 (16)  |           |           |           |           |           |           |           |  |
| gFrontWindow      | 2676*(13) | 601=(14)  | 164 (16)  | 779 (16)  | 781 (16)  | 781 (16)  | 783 (16)  | 783 (16)  |  |
|                   | 937 (16)  | 1576 (16) | 1816 (16) | 1816 (16) | 1826 (16) | 1826 (16) | 2897 (16) | 2901 (16) |  |
|                   | 2905 (16) | 3012 (16) | 3032 (16) | 269 (19)  | 271 (19)  | 272=(19)  | 280=(19)  | 381 (19)  |  |
|                   | 382=(19)  | 896 (19)  |           |           |           |           |           |           |  |
| gGNEPatch         | 39*( 4)   | 154 ( 4)  | 192 ( 4)  |           |           |           |           |           |  |
| gGotClipType      | 2678*(13) | 78=(14)   | 661=(14)  | 938 (16)  | 1096 (16) | 2852=(16) | 2914 (16) |           |  |
| gGZPurgeNotify    | 29*(28)   | 423=(28)  | 674 (28)  | 675 (28)  |           |           |           |           |  |
| gHeadCohandler    | 532 (16)  |           |           |           |           |           |           |           |  |
| gHeadCohandler    | 2679*(13) | 642=(14)  | 939 (16)  | 1344 (16) | 1794 (16) | 1841 (16) | 1842=(16) | 1847 (16) |  |
|                   | 1853=(16) |           |           |           |           |           |           |           |  |
| gHNullMenuProc    | 168*(29)  | 340=(30)  | 341 (30)  | 342 (30)  | 393 (30)  | 448 (30)  |           |           |  |
| gICPatch          | 40*( 4)   | 144 ( 4)  | 186 ( 4)  |           |           |           |           |           |  |
| gIdleFreq         | 2681*(13) | 577=(14)  | 940 (16)  | 1768=(16) | 1768 (16) | 1788=(16) | 1791 (16) | 1793=(16) |  |
|                   | 2355 (16) | 2793=(16) |           |           |           |           |           |           |  |
| gIdlePhase        | 2683*(13) | 574=(14)  | 941 (16)  | 1154 (16) | 1157=(16) | 1189 (16) | 1904=(16) | 1906 (16) |  |
|                   | 2357 (16) | 2360=(16) | 2371 (16) | 2372=(16) |           |           |           |           |  |
| gInBackground     | 2684*(13) | 575=(14)  | 942 (16)  | 1178 (16) | 1692=(16) | 3012 (16) |           |           |  |
| gInitHandler      | 12*( 8)   | 176 ( 8)  | 178 ( 8)  | 179=( 8)  | 202=( 8)  | 286=( 8)  |           |           |  |
| gInitialized      | 2685*(13) | 382=(14)  | 785 (14)  | 943 (16)  | 2689=(16) | 1628 (37) |           |           |  |
| gInspectLinePos   | 12*(32)   | 164 (32)  | 167=(32)  | 169 (32)  | 172=(32)  | 172 (32)  | 175=(32)  | 175 (32)  |  |
|                   | 257=(32)  | 259=(32)  | 309=(32)  |           |           |           |           |           |  |
| gIntenseDebugging | 231 ( 6)  | 223*(13)  | 2688*(13) | 431=(14)  | 502 (14)  | 145 (15)  | 772=(16)  | 772 (16)  |  |
|                   | 944 (16)  | 1140 (16) | 2894 (16) | 3021 (16) | 3077 (16) | 977 (18)  | 493 (20)  | 114 (23)  |  |
|                   | 71 (39)   | 337 (40)  | 1309 (40) | 1323 (40) |           |           |           |           |  |
| GivePasteData     | 1665*(13) | 1100 (16) | 1237*(18) | 1256=(18) | 259*(38)  | 1076*(40) | 1099=(40) | 1102=(40) |  |
|                   | 1170=(40) | 491 (41)  |           |           |           |           |           |           |  |
| gLastClickPart    | 2690*(13) | 573=(14)  | 589 (16)  | 598=(16)  | 945 (16)  | 2738=(16) |           |           |  |
| gLastCommand      | 2692*(13) | 98=(16)   | 156 (16)  | 219 (16)  | 219 (16)  | 384 (16)  | 564 (16)  | 566 (16)  |  |
|                   | 567 (16)  | 568 (16)  | 569=(16)  | 800 (16)  | 803 (16)  | 805 (16)  | 810 (16)  | 814 (16)  |  |
|                   | 814 (16)  | 816 (16)  | 818 (16)  | 946 (16)  | 2264=(16) | 2277 (16) | 2297=(16) | 2873 (16) |  |
|                   | 2874 (16) | 208 (17)  | 209 (17)  | 987 (17)  | 988 (17)  | 1106 (17) | 1107 (17) |           |  |
| gLastDeskAcc      | 2694*(13) | 576=(14)  | 213=(16)  | 947 (16)  | 522 (37)  |           |           |           |  |
| gLastIdle         | 2696*(13) | 578=(14)  | 948 (16)  | 1787=(16) | 1791 (16) | 1796=(16) |           |           |  |
| gLastMsePt        | 2698*(13) | 592 (16)  | 595=(16)  | 949 (16)  |           |           |           |           |  |
| gLastUpTime       | 2700*(13) | 572=(14)  | 576 (14)  | 589 (14)  | 591 (16)  | 950 (16)  | 1648=(16) | 792=(19)  |  |
| global            | 553*(19)  | 562 (19)  | 565 (19)  | 593 (19)  | 594 (19)  | 613 (19)  | 614 (19)  |           |  |
| globalBounds      | 2018*(13) | 360 (19)  | 363 (19)  | 365 (19)  | 626*(19)  | 636=(19)  | 637 (19)  | 638 (19)  |  |
| globalbounds      | 350*(19)  | 359 (19)  |           |           |           |           |           |           |  |
| globalMce         | 152*( 6)  | 186=( 6)  | 204 ( 6)  | 205 ( 6)  | 217 ( 6)  | 222 ( 6)  |           |           |  |
| globalMouse       | 741*(13)  | 1963*(13) | 1965*(13) | 3001*(16) | 3015 (16) | 3016 (16) | 3018 (16) | 3027 (16) |  |
|                   | 3044 (16) | 3045 (16) | 3073 (16) | 3098*(16) | 3248 (16) | 784*(19)  | 791 (19)  | 848*(19)  |  |
|                   | 856 (19)  |           |           |           |           |           |           |           |  |
| GlobalToLocal     | 1611 (16) | 3251 (16) | 3341 (16) |           |           |           |           |           |  |
| gLongOffset       | 2702*(13) | 291 (14)  | 1197=(14) | 951 (16)  | 16 (18)   | 892=(18)  | 921 (18)  | 924 (18)  |  |
|                   | 1037 (18) | 1037 (18) | 1041 (18) | 1042 (18) | 1519 (18) | 1529 (18) | 1529 (18) | 1795 (18) |  |
|                   | 1805 (18) | 1805 (18) | 521=(19)  | 281=(20)  | 321=(20)  | 497 (20)  | 943 (37)  | 946 (37)  |  |
| gLowSpaceInterval | 2703*(13) | 590=(14)  | 952 (16)  | 2963 (16) | 2969 (16) |           |           |           |  |
| gMainEventMask    | 2708*(13) | 511=(14)  | 953 (16)  | 2355 (16) |           |           |           |           |  |
| gMainFileType     | 2710*(13) | 95=(16)   | 954 (16)  | 2939 (16) |           |           |           |           |  |
| gMastReport       | 433=(14)  | 503 (14)  |           |           |           |           |           |           |  |
| gMaxLockedRsrc    | 160*(27)  | 306 (28)  | 308=(28)  | 311 (28)  | 369=(28)  |           |           |           |  |
| gMBarDisplayed    | 2715*(13) | 674=(14)  | 104 (16)  | 113 (16)  | 955 (16)  |           |           |           |  |
| gMBarHeight       | 2717*(13) | 610=(14)  | 614=(14)  | 616=(14)  | 620 (14)  | 624 (14)  | 626 (14)  | 956 (16)  |  |
|                   | 589 (19)  |           |           |           |           |           |           |           |  |
| gMBarHierarchical | 2718*(13) | 676=(14)  | 133 (16)  | 957 (16)  |           |           |           |           |  |
| gMBarNotDisplayed | 2720*(13) | 675=(14)  | 128 (16)  | 958 (16)  |           |           |           |           |  |
| gMemMgtBreak      | 435=(14)  | 498 (14)  | 162*(27)  | 313 (28)  | 779 (28)  | 1143 (28) |           |           |  |
| gMemMgtReport     | 227*(13)  |           |           |           |           |           |           |           |  |
| gMemReserve       | 33*(28)   | 209 (28)  | 210 (28)  | 253 (28)  | 254 (28)  | 343 (28)  | 383 (28)  | 419=(28)  |  |
|                   | 690 (28)  | 692 (28)  | 841 (28)  | 1000 (28) | 1159 (28) |           |           |           |  |
| gMenusAreSetup    | 959 (16)  | 2400=(16) | 2918 (16) | 156*(29)  | 508=(30)  |           |           |           |  |
| gNewScrapStuff    | 2721*(13) | 243 (16)  | 245=(16)  | 382 (16)  | 392 (16)  | 191 (23)  | 266 (23)  | 271 (23)  |  |
| gNextSpaceMsg     | 2722*(13) | 589=(16)  | 960 (16)  | 1784=(16) | 2966 (16) | 2969=(16) |           |           |  |
| gNoChanges        | 483 ( 6)  | 581 ( 6)  | 604 ( 6)  | 2254 ( 6) | 3027 ( 6) | 3048 ( 6) | 3075 ( 6) | 3080 ( 6) |  |
|                   | 675 (10)  | 2724*(13) | 597=(14)  | 206 (15)  | 238 (15)  | 254 (15)  | 630 (16)  | 672 (16)  |  |
|                   | 719 (16)  | 961 (16)  | 1321 (16) | 1343 (16) | 1371 (16) | 1417 (16) | 1561 (16) | 1616 (16) |  |
|                   | 1649 (16) | 1696 (16) | 1722 (16) | 1971 (16) | 2273 (16) | 2461 (16) | 3176 (16) | 3224 (16) |  |
|                   | 3238 (16) | 314 (17)  | 805 (18)  | 1323 (18) | 694 (19)  | 83 (21)   | 683 (21)  | 1074 (21) |  |
|                   | 1374 (21) | 163 (24)  | 659 (37)  | 1142 (37) | 1330 (37) | 1543 (37) | 1932 (37) | 701 (40)  |  |
|                   | 837 (40)  | 893 (40)  |           |           |           |           |           |           |  |

|                     |            |            |            |            |            |            |            |           |
|---------------------|------------|------------|------------|------------|------------|------------|------------|-----------|
| gNonResidents       | 494 (14)   | 191* (27)  | 461- (28)  | 462 (28)   | 468- (28)  | 1018- (28) | 1024- (28) | 1134 (28) |
| gNoOfSegments       | 34* (28)   | 446- (28)  | 454- (28)  | 454 (28)   | 459 (28)   | 464 (28)   | 465 (28)   | 1131 (28) |
| gNullPrintHandler   | 2728* (13) | 645 (14)   | 646 (14)   | 647 (14)   | 649 (14)   | 86 (16)    | 962 (16)   | 126 (24)  |
|                     | 197 (37)   | 325 (37)   |            |            |            |            |            |           |
| gNumUntitled        | 2730* (13) | 666- (14)  | 963 (16)   | 1471 (17)  | 1474- (17) | 1474 (17)  |            |           |
| goAway              | 1219* (16) | 1269 (16)  |            |            |            |            |            |           |
| goAwayFlag          | 865 (16)   |            |            |            |            |            |            |           |
| gObjNameTable       | 11* (32)   | 50- (32)   | 85 (32)    | 112 (32)   | 115 (32)   |            |            |           |
| gOKCodeReserve      | 35* (28)   | 230- (28)  | 270- (28)  | 290 (28)   | 337 (28)   | 382 (28)   | 424- (28)  | 735 (28)  |
| gOldChooserFlag     | 2736* (13) | 112 (14)   | 546- (14)  | 964 (16)   |            |            |            |           |
| gOldScrapStuff      | 2738* (13) | 243- (16)  | 382 (16)   |            |            |            |            |           |
| good                | 454 (16)   | 455 (16)   | 930 (17)   |            |            |            |            |           |
| gOrthogonal         | 2739* (13) | 629- (14)  | 630- (14)  | 965 (16)   | 966 (16)   | 163 (20)   | 167 (20)   | 415 (37)  |
|                     | 458 (37)   | 757 (37)   | 783 (37)   | 850 (37)   | 851 (37)   | 978 (37)   | 994 (37)   | 1233 (37) |
|                     | 1855 (37)  | 1856 (37)  | 2038 (37)  | 585 (40)   |            |            |            |           |
| gotATracker         | 3104* (16) | 3176- (16) | 3177 (16)  | 3191 (16)  | 3204 (16)  | 3225- (16) | 3229 (16)  | 3293 (16) |
| gotCode             | 305* (27)  | 327* (28)  | 382- (28)  |            |            |            |            |           |
| gotLowSpace         | 305* (27)  | 327* (28)  | 383- (28)  |            |            |            |            |           |
| gotTitle            | 148* (6)   | 165- (6)   | 178- (6)   | 182 (6)    | 197 (6)    | 214 (6)    | 219 (6)    |           |
| gPageOffset         | 2740* (13) | 967 (16)   | 943 (37)   | 947 (37)   | 948 (37)   | 949 (37)   | 1914 (37)  | 942 (40)  |
|                     | 942 (40)   |            |            |            |            |            |            |           |
| gPatchList          | 40* (35)   | 53- (35)   | 86 (35)    | 87- (35)   | 219 (35)   | 220- (35)  | 223 (35)   | 259 (35)  |
|                     | 260 (35)   |            |            |            |            |            |            |           |
| gPermAllocation     | 38* (28)   | 202 (28)   | 203- (28)  | 213- (28)  | 235 (28)   | 418- (28)  | 628 (28)   | 672 (28)  |
|                     | 682 (28)   | 872 (28)   | 874 (28)   | 876- (28)  | 906 (28)   |            |            |           |
| gPPrPort            | 39* (37)   | 1155- (37) | 1162 (37)  | 1182 (37)  | 1589 (37)  | 1595 (37)  | 1601 (37)  | 1609 (37) |
| gPrefClipType       | 2742* (13) | 79- (14)   | 968 (16)   | 1098 (16)  |            |            |            |           |
| gPrintDialog        | 521* (36)  | 73 (37)    | 74 (37)    | 78 (37)    | 179- (37)  | 395 (37)   | 397 (37)   | 398- (37) |
|                     | 1392- (37) | 1398 (37)  | 1408 (37)  | 1416 (37)  | 1417 (37)  | 1997- (37) | 1998 (37)  | 2001 (37) |
|                     | 2002 (37)  |            |            |            |            |            |            |           |
| gPrintHandler       | 2743* (13) | 649- (14)  | 163 (16)   | 531 (16)   | 969 (16)   | 197 (37)   | 203- (37)  |           |
| gPrinting           | 2749* (13) | 632- (14)  | 1138 (14)  | 970 (16)   | 907 (18)   | 1159 (18)  | 93 (37)    | 94- (37)  |
|                     | 98- (37)   | 1164- (37) | 1178- (37) | 988 (40)   |            |            |            |           |
| gProtoList          | 2751* (13) | 692 (14)   | 692 (14)   | 692 (14)   | 1159 (14)  | 1160 (14)  | 73 (15)    | 972 (16)  |
| GrafPort            | 1836 (18)  |            |            |            |            |            |            |           |
| GrafPtr             | 1550 (13)  | 1639 (13)  | 2016 (13)  | 2168 (13)  | 2810 (13)  | 3124 (16)  | 337 (18)   | 533 (18)  |
|                     | 1157 (18)  | 629 (19)   | 662 (19)   | 665 (19)   | 602 (21)   | 627 (21)   | 57 (37)    | 1162 (37) |
|                     | 1572 (37)  | 1601 (37)  | 227 (38)   | 214 (40)   | 302 (40)   | 415 (40)   | 984 (40)   | 985 (40)  |
|                     | 1206 (40)  | 1364 (40)  |            |            |            |            |            |           |
| gray                | 1219 (14)  | 279 (21)   |            |            |            |            |            |           |
| GrayRgn             | 550* (19)  | 586 (19)   |            |            |            |            |            |           |
| gRedrawMenuBar      | 974 (16)   | 2397 (16)  | 2418- (16) | 2708- (16) | 2863- (16) | 2918 (16)  | 162* (29)  | 485- (30) |
|                     | 502 (30)   | 505- (30)  |            |            |            |            |            |           |
| green               | 99- (6)    | 123 (6)    | 234 (6)    | 237 (6)    | 342* (25)  | 320- (26)  | 794 (26)   | 829* (26) |
|                     | 833- (26)  | 833 (26)   | 1019 (26)  | 1021 (26)  | 1026 (26)  |            |            |           |
| greenColor          | 130 (6)    | 801 (26)   |            |            |            |            |            |           |
| gReportEvt          | 225* (13)  | 2754* (13) | 432- (14)  | 500 (14)   | 766- (16)  | 766 (16)   | 975 (16)   | 1140 (16) |
|                     | 1172 (16)  | 1409 (16)  | 1569 (16)  | 2891 (16)  |            |            |            |           |
| gReportInfo         | 777 (28)   | 778- (28)  | 778 (28)   |            |            |            |            |           |
| gReportMenuChoices  | 222* (13)  | 2755* (13) | 430- (14)  | 499 (14)   | 770- (16)  | 770 (16)   | 976 (16)   | 1988 (16) |
|                     | 2893 (16)  |            |            |            |            |            |            |           |
| gReportNext         | 775- (28)  |            |            |            |            |            |            |           |
| gReserveExists      | 39* (28)   | 239 (28)   | 241- (28)  | 271- (28)  | 425- (28)  | 632- (28)  | 656- (28)  | 680- (28) |
|                     | 999- (28)  |            |            |            |            |            |            |           |
| gReserveShortfall   | 42* (28)   | 233- (28)  | 267- (28)  | 380 (28)   |            |            |            |           |
| gRGBBlack           | 93 (6)     | 160 (6)    | 2404 (6)   | 414 (14)   | 687 (14)   | 205* (25)  | 314 (26)   | 747 (37)  |
|                     | 366 (41)   |            |            |            |            |            |            |           |
| gRGBWhite           | 161 (6)    | 415 (14)   | 206* (25)  |            |            |            |            |           |
| GridCell            | 75* (9)    | 240 (9)    | 244 (9)    | 252 (9)    | 255 (9)    | 262 (9)    | 265 (9)    | 294 (9)   |
|                     | 295 (9)    | 299 (9)    | 303 (9)    | 312 (9)    | 343 (9)    | 350 (9)    | 353 (9)    | 356 (9)   |
|                     | 366 (9)    | 388 (9)    | 469 (9)    | 479 (9)    | 565 (9)    | 588 (9)    | 603 (9)    | 673 (9)   |
|                     | 678 (9)    | 682 (9)    | 309 (10)   | 311 (10)   | 312 (10)   | 367 (10)   | 381 (10)   | 395 (10)  |
|                     | 494 (10)   | 629 (10)   | 693 (10)   | 694 (10)   | 767 (10)   | 772 (10)   | 823 (10)   | 1081 (10) |
|                     | 1082 (10)  | 1089 (10)  | 1125 (10)  | 1139 (10)  | 1143 (10)  | 1330 (10)  | 1404 (10)  | 1453 (10) |
|                     | 1762 (10)  | 1773 (10)  | 1841 (10)  | 1854 (10)  | 1933 (10)  | 2072 (10)  | 2075 (10)  | 2315 (10) |
|                     | 2389 (10)  | 2496 (10)  | 2574 (10)  | 2627 (10)  | 2679 (10)  | 2709 (10)  | 2723 (10)  | 2854 (10) |
|                     | 2870 (10)  | 2933 (10)  | 2934 (10)  |            |            |            |            |           |
| GridViewPart        | 76* (9)    | 320 (9)    | 623 (10)   | 1449 (10)  |            |            |            |           |
| GridViewTemplate    | 92* (9)    | 104 (9)    | 229 (10)   | 249 (10)   |            |            |            |           |
| GridViewTemplatePtr | 104* (9)   | 148 (10)   | 243 (10)   | 248 (10)   |            |            |            |           |
| grievance           | 230* (7)   | 233* (7)   | 219* (8)   | 251 (8)    | 268* (8)   | 271 (8)    |            |           |
| growResult          | 851* (19)  | 856- (19)  | 857 (19)   | 858 (19)   | 858 (19)   |            |            |           |
| GrowWindow          | 856 (19)   |            |            |            |            |            |            |           |
| GrowZoneProc        | 54* (28)   | 436 (28)   | 615* (28)  | 696- (28)  |            |            |            |           |
| gRsrcCheck          | 2756* (13) | 541- (14)  | 800- (14)  | 444- (16)  | 977 (16)   | 1129- (16) | 1129 (16)  | 1130 (16) |
|                     | 1133- (16) | 921- (17)  |            |            |            |            |            |           |
| gRsrcReport         | 434- (14)  | 505 (14)   | 164* (27)  | 309 (28)   |            |            |            |           |
| gSaveClip           | 2762* (13) | 292 (14)   | 569- (14)  | 1196 (14)  | 978 (16)   |            |            |           |
| gSaveFocusedView    | 2763* (13) | 293- (14)  | 1198 (14)  | 979 (16)   |            |            |            |           |
| gSaveLongOffset     | 2765* (13) | 291- (14)  | 1197 (14)  | 980 (16)   |            |            |            |           |
| gSaveOrg            | 2764* (13) | 290- (14)  | 1195 (14)  | 1195 (14)  | 981 (16)   |            |            |           |
| gSavePort           | 2766* (13) | 289 (14)   | 1194 (14)  | 982 (16)   |            |            |            |           |
| gSCCPatch           | 42* (4)    | 146 (4)    | 188 (4)    |            |            |            |            |           |
| gSCPatch            | 41* (4)    | 143 (4)    | 187 (4)    |            |            |            |            |           |
| gSegLoadPatch       | 49* (28)   | 516 (28)   | 765 (28)   |            |            |            |            |           |
| gSegNeedsUnloading  | 23* (28)   | 464- (28)  | 466- (28)  | 469- (28)  | 769- (28)  | 1019- (28) | 1025- (28) | 1132 (28) |
|                     | 1138- (28) |            |            |            |            |            |            |           |
| gSegReport          | 506 (14)   | 165* (27)  | 427- (28)  | 772 (28)   | 1142 (28)  |            |            |           |
| gSignatureCount     | 2771* (13) | 259 (14)   | 690- (14)  | 1173- (14) | 1173 (14)  | 1175 (14)  | 1178 (14)  | 1179 (14) |
|                     | 984 (16)   |            |            |            |            |            |            |           |
| gSignatures         | 2769* (13) | 260 (14)   | 1178- (14) |            |            |            |            |           |
| gSignatureViews     | 2770* (13) | 262 (14)   | 1179- (14) |            |            |            |            |           |
| gSingleStep         | 779- (28)  |            |            |            |            |            |            |           |
| gSizeCmdTable       | 26* (30)   | 72 (30)    | 116 (30)   | 325- (30)  | 329 (30)   |            |            |           |
| gStdHysteresis      | 2774* (13) | 599- (14)  | 986 (16)   | 1613 (16)  |            |            |            |           |
| gStdPageMargins     | 518* (36)  | 182 (37)   | 281 (37)   |            |            |            |            |           |
| gStdStaggerCount    | 2775* (13) | 672- (14)  | 987 (16)   | 156 (19)   |            |            |            |           |
| gStdMoveBounds      | 2776* (13) | 620 (14)   | 988 (16)   | 63 (19)    | 138 (19)   |            |            |           |
| gStdWScreenRect     | 2780* (13) | 626 (14)   | 990 (16)   | 588 (19)   | 589 (19)   | 597 (19)   | 598 (19)   | 599 (19)  |



|                     |            |            |            |            |            |            |            |            |
|---------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| gStdWSizeRect       | 600 (19)   | 602 (19)   | 603 (19)   | 604 (19)   | 605 (19)   |            |            |            |
| gSysMemList         | 2778* (13) | 624 (14)   | 989 (16)   | 64 (19)    | 64 (19)    | 139 (19)   | 139 (19)   |            |
| gSystemStyle        | 167* (27)  | 472- (28)  | 473 (28)   | 474 (28)   | 475 (28)   | 476 (28)   | 477 (28)   | 979 (28)   |
|                     | 620 ( 6)   | 2936 ( 6)  | 2786* (13) | 687 (14)   | 992 (16)   | 993 (16)   | 994 (16)   | 995 (16)   |
|                     | 996 (16)   | 105 (21)   | 116 (21)   | 89 (23)    | 242 (23)   |            |            |            |
| gSysWindowActive    | 2785* (13) | 78- (16)   | 991 (16)   | 1964 (16)  | 2007 (16)  | 2012 (16)  | 2377 (16)  | 2411 (16)  |
|                     | 2413- (16) | 2415 (16)  | 2868 (16)  | 2913 (16)  |            |            |            |            |
| gSzCodeReserve      | 44* (28)   | 247 (28)   | 336 (28)   | 355 (28)   | 379 (28)   | 422- (28)  | 553 (28)   | 654 (28)   |
|                     | 995- (28)  |            |            |            |            |            |            |            |
| gSzMemReserve       | 46* (28)   | 210 (28)   | 342 (28)   | 362 (28)   | 381 (28)   | 420- (28)  | 554 (28)   | 996- (28)  |
| gTarget             | 3068 ( 6)  | 3072 ( 6)  | 2787* (13) | 786 (14)   | 1027 (14)  | 26 (15)    | 28- (15)   | 30- (15)   |
|                     | 79- (16)   | 155 (16)   | 592 (16)   | 776 (16)   | 997 (16)   | 1537 (16)  | 1539 (16)  | 1795 (16)  |
|                     | 2010 (16)  | 2285 (16)  | 2349 (16)  | 2790 (16)  | 2792- (16) | 2854 (16)  | 401 (17)   | 278 (19)   |
| gTEIntenseDebugging | 11* (39)   | 24- (40)   | 77- (40)   | 269 (41)   | 289 (41)   | 306 (41)   | 403 (41)   | 573 (41)   |
|                     | 590 (41)   | 863 (41)   | 884 (41)   | 899 (41)   | 926 (41)   |            |            |            |
| gTempRgn            | 2796* (13) | 133 (14)   | 568- (14)  | 998 (16)   | 3058 (16)  | 3059 (16)  | 163 (18)   | 164 (18)   |
|                     | 164 (18)   | 165 (18)   | 587 (18)   | 588 (18)   | 588 (18)   | 589 (18)   | 590 (18)   | 1590 (18)  |
|                     | 1592 (18)  | 1592 (18)  | 1593 (18)  | 486 (19)   | 490 (19)   | 587 (19)   | 588 (19)   | 593 (19)   |
|                     | 509 (20)   | 510 (20)   | 1000 (40)  | 1003 (40)  | 1016 (40)  | 1019 (40)  |            |            |
| gTopHandler         | 10* ( 8)   | 153 ( 8)   | 171- ( 8)  | 201- ( 8)  | 300 ( 8)   | 303 ( 8)   | 311- ( 8)  |            |
| gTraceIdle          | 2798* (13) | 493- (14)  | 790- (16)  | 790 (16)   | 1000 (16)  | 1779 (16)  | 2910 (16)  | 3021 (16)  |
|                     | 3077 (16)  |            |            |            |            |            |            |            |
| gTraceSetupMenus    | 492- (14)  | 788- (16)  | 788 (16)   | 2909 (16)  | 170* (29)  | 249 (30)   | 274 (30)   | 523 (30)   |
|                     | 545 (30)   | 602 (30)   |            |            |            |            |            |            |
| gTracing            | 249 (30)   | 274 (30)   | 523 (30)   | 545 (30)   | 602 (30)   |            |            |            |
| gUndoCmd            | 2801* (13) | 669- (14)  | 1003 (16)  | 2809 (16)  | 2831- (16) |            |            |            |
| gUndoState          | 2800* (13) | 668- (14)  | 1002 (16)  | 2809 (16)  | 2830- (16) |            |            |            |
| gUnloadAllSegs      | 507 (14)   | 145* (27)  | 195* (27)  | 428- (28)  | 1124 (28)  |            |            |            |
| gUsedBy             | 2803* (13) | 132- (14)  | 594- (14)  | 1275 (14)  | 1281- (14) | 1004 (16)  |            |            |
| gVarClipPicSize     | 2805* (13) | 100- (16)  | 1005 (16)  | 122 (23)   | 252 (23)   |            |            |            |
| gVPtr               | 243* (10)  | 248- (10)  | 251 (10)   |            |            |            |            |            |
| gWorkPort           | 2810* (13) | 582- (14)  | 584 (14)   | 586 (14)   | 1006 (16)  | 610 (21)   | 639 (21)   | 649 (21)   |
|                     | 221 (40)   | 321 (40)   | 445 (40)   | 1217 (40)  | 1406 (40)  |            |            |            |
| gWResSignature      | 369- ( 6)  | 846- ( 6)  | 926- ( 6)  | 1053- ( 6) | 1202- ( 6) | 1511- ( 6) | 1730- ( 6) | 1930- ( 6) |
|                     | 2167- ( 6) | 2759- ( 6) | 2997- ( 6) | 3466- ( 6) | 279- (10)  | 2304- (10) | 2474- (10) | 2812* (13) |
|                     | 1007 (16)  | 260 (18)   | 277- (18)  | 216- (19)  | 113- (20)  | 206- (21)  | 1019- (21) | 166- (40)  |
| gWResType           | 369- ( 6)  | 846- ( 6)  | 926- ( 6)  | 1053- ( 6) | 1202- ( 6) | 1511- ( 6) | 1730- ( 6) | 1930- ( 6) |
|                     | 2167- ( 6) | 2759- ( 6) | 2997- ( 6) | 3466- ( 6) | 279- (10)  | 2304- (10) | 2474- (10) | 2813* (13) |
|                     | 1008 (16)  | 245 (18)   | 262 (18)   | 277- (18)  | 216- (19)  | 113- (20)  | 206- (21)  | 1019- (21) |
|                     | 166- (40)  |            |            |            |            |            |            |            |
| gWrToWindow         | 16 (26)    |            |            |            |            |            |            |            |
| gWWPerLine          | 164 (32)   |            |            |            |            |            |            |            |
| gZeroRect           | 620 ( 6)   | 2814* (13) | 603 (14)   | 604 (14)   | 1009 (16)  | 1055 (18)  | 114 (21)   | 910 (21)   |
|                     | 1001 (40)  | 1017 (40)  |            |            |            |            |            |            |
| gZeroVPt            | 619 ( 6)   | 619 ( 6)   | 2815* (13) | 605 (14)   | 858 (14)   | 940 (14)   | 1010 (16)  | 2767 (16)  |
|                     | 1138 (18)  | 1182 (18)  | 41 (19)    | 41 (19)    | 111 (19)   | 521 (19)   | 26 (20)    | 27 (20)    |
|                     | 59 (20)    | 192 (20)   | 541 (20)   | 560 (20)   | 278 (37)   | 375 (41)   |            |            |
| gZeroVRect          | 2816* (13) | 606 (14)   | 924 (14)   | 1011 (16)  | 1641 (18)  | 28 (20)    |            |            |
| GZMoveHand          | 83* (28)   | 713 (28)   |            |            |            |            |            |            |
| GZRootHand          | 92* (28)   | 713 (28)   |            |            |            |            |            |            |
| -H-                 |            |            |            |            |            |            |            |            |
| h                   | 604* ( 5)  | 1261 ( 6)  | 1269 ( 6)  | 2272 ( 6)  | 2592 ( 6)  | 2594 ( 6)  | 3218* ( 6) | 3221 ( 6)  |
|                     | 3223 ( 6)  | 3346- ( 6) | 3346 ( 6)  | 99- (10)   | 219 (10)   | 220- (10)  | 220 (10)   | 315- (10)  |
|                     | 317- (10)  | 357- (10)  | 400 (10)   | 401 (10)   | 405 (10)   | 415 (10)   | 496 (10)   | 497 (10)   |
|                     | 634- (10)  | 728 (10)   | 728 (10)   | 796 (10)   | 796 (10)   | 804- (10)  | 1093 (10)  | 1095 (10)  |
|                     | 1113- (10) | 1461 (10)  | 1463 (10)  | 1464 (10)  | 1474 (10)  | 1479 (10)  | 1869- (10) | 1936 (10)  |
|                     | 1939 (10)  | 1939 (10)  | 2078 (10)  | 2079- (10) | 2087- (10) | 2089- (10) | 2089 (10)  | 2094- (10) |
|                     | 2095- (10) | 2095 (10)  | 2096 (10)  | 2098- (10) | 2098 (10)  | 2100- (10) | 2100 (10)  | 2104- (10) |
|                     | 2104 (10)  | 2104 (10)  | 2221 (10)  | 2223 (10)  | 2254 (10)  | 2256 (10)  | 2321 (10)  | 2461 (10)  |
|                     | 2682- (10) | 2725- (10) | 2891 (10)  | 2893 (10)  | 2898 (10)  | 2898 (10)  | 2900 (10)  | 2900 (10)  |
|                     | 2944 (10)  | 2945 (10)  | 2956 (10)  | 459 (13)   | 1574* (13) | 1813* (13) | 1841* (13) | 1961* (13) |
|                     | 331* (14)  | 629 (14)   | 630 (14)   | 862 (14)   | 884 (14)   | 934- (14)  | 934 (14)   | 961 (14)   |
|                     | 1195 (14)  | 267 (15)   | 267 (15)   | 965 (16)   | 2573 (16)  | 3044- (16) | 3044 (16)  | 3074 (16)  |
|                     | 3074 (16)  | 3163 (16)  | 3163 (16)  | 3284 (16)  | 3284 (16)  | 3310 (16)  | 3318 (16)  | 196 (18)   |
|                     | 257 (18)   | 383 (18)   | 383 (18)   | 385 (18)   | 451 (18)   | 471 (18)   | 481 (18)   | 499 (18)   |
|                     | 499 (18)   | 507 (18)   | 511 (18)   | 630 (18)   | 890 (18)   | 896 (18)   | 1033 (18)  | 1045 (18)  |
|                     | 1047 (18)  | 1149 (18)  | 1429* (18) | 1440 (18)  | 1440 (18)  | 1444- (18) | 1444 (18)  | 1529 (18)  |
|                     | 1568 (18)  | 1570 (18)  | 1572- (18) | 1630 (18)  | 1717 (18)  | 1729 (18)  | 1805 (18)  | 1912 (18)  |
|                     | 110 (19)   | 110 (19)   | 110 (19)   | 325 (19)   | 363 (19)   | 363 (19)   | 761* (19)  | 765 (19)   |
|                     | 822 (19)   | 831 (19)   | 835 (19)   | 882 (19)   | 988- (19)  | 988 (19)   | 991 (19)   | 21 (20)    |
|                     | 32- (20)   | 52 (20)    | 60- (20)   | 93 (20)    | 96 (20)    | 98 (20)    | 100 (20)   | 125 (20)   |
|                     | 126 (20)   | 158 (20)   | 172 (20)   | 173 (20)   | 194 (20)   | 225 (20)   | 226 (20)   | 230 (20)   |
|                     | 230 (20)   | 233 (20)   | 237 (20)   | 237 (20)   | 261 (20)   | 268 (20)   | 304 (20)   | 307 (20)   |
|                     | 332 (20)   | 374* (20)  | 377 (20)   | 406 (20)   | 419 (20)   | 441 (20)   | 450 (20)   | 467 (20)   |
|                     | 468 (20)   | 502 (20)   | 509 (20)   | 592* (20)  | 595 (20)   | 595 (20)   | 612 (20)   | 626 (20)   |
|                     | 626 (20)   | 628 (20)   | 641- (20)  | 643 (20)   | 101 (21)   | 234- (21)  | 234 (21)   | 234 (21)   |
|                     | 264 (21)   | 385 (21)   | 443 (21)   | 541 (21)   | 949 (21)   | 979 (21)   | 102- (23)  | 121 (23)   |
|                     | 257 (23)   | 160* (25)  | 258* (25)  | 332* (25)  | 274* (26)  | 279 (26)   | 282 (26)   | 677 (26)   |
|                     | 677- (26)  | 678 (26)   | 678- (26)  | 764* (26)  | 769- (26)  | 772 (26)   | 982 (26)   | 1047 (26)  |
|                     | 1175 (26)  | 1273 (26)  | 264* (27)  | 273* (27)  | 276* (27)  | 56* (28)   | 73* (28)   | 105* (28)  |
|                     | 115- (28)  | 116 (28)   | 121- (28)  | 123 (28)   | 124 (28)   | 136* (28)  | 140 (28)   | 302* (28)  |
|                     | 305 (28)   | 414* (28)  | 486- (28)  | 487 (28)   | 488 (28)   | 493- (28)  | 494 (28)   | 500 (28)   |
|                     | 708* (28)  | 710 (28)   | 713 (28)   | 713 (28)   | 935* (28)  | 939 (28)   | 939 (28)   | 948* (28)  |
|                     | 1067* (28) | 1071 (28)  | 1072 (28)  | 1074 (28)  | 1076 (28)  | 1079 (28)  | 1084 (28)  | 1085 (28)  |
|                     | 268 (37)   | 282- (37)  | 456 (37)   | 484 (37)   | 543- (37)  | 555 (37)   | 555 (37)   | 557 (37)   |
|                     | 557 (37)   | 755 (37)   | 791 (37)   | 792 (37)   | 797 (37)   | 886 (37)   | 941 (37)   | 953 (37)   |
|                     | 1110 (37)  | 1256 (37)  | 1853 (37)  | 1888 (37)  | 1899 (37)  | 1913 (37)  | 195 (39)   | 201 (39)   |
|                     | 215 (39)   | 217 (39)   | 283 (40)   | 285 (40)   | 380 (40)   | 381- (40)  | 381 (40)   | 518 (40)   |
|                     | 521 (40)   | 533 (40)   | 596 (40)   | 940 (40)   | 1221 (40)  | 1269 (40)  | 1279 (40)  | 1279 (40)  |
|                     | 1320 (40)  | 1466* (40) | 1468- (40) | 1469 (40)  | 1476 (40)  | 16* (41)   | 55- (41)   | 56 (41)    |
|                     | 59 (41)    | 61 (41)    | 39* (42)   |            |            |            |            |            |
| Handle              | 116 ( 5)   | 303 ( 5)   | 327 ( 5)   | 349 ( 5)   | 373 ( 5)   | 510 ( 5)   | 530 ( 5)   | 593 ( 5)   |
|                     | 667 ( 6)   | 1275 ( 6)  | 1282 ( 6)  | 1286 ( 6)  | 1289 ( 6)  | 1291 ( 6)  | 1413 ( 6)  | 1459 ( 6)  |
|                     | 1577 ( 6)  | 1636 ( 6)  | 1640 ( 6)  | 1682 ( 6)  | 1686 ( 6)  | 1797 ( 6)  | 1960 ( 6)  | 1965 ( 6)  |
|                     | 1966 ( 6)  | 1969 ( 6)  | 2053 ( 6)  | 2109 ( 6)  | 2447 ( 6)  | 2448 ( 6)  | 2505 ( 6)  | 2779 ( 6)  |
|                     | 2808 ( 6)  | 2885 ( 6)  | 3102 ( 6)  | 3154 ( 6)  | 891 (10)   | 902 (10)   | 988 (10)   | 999 (10)   |
|                     | 1559 (10)  | 1570 (10)  | 1573 (10)  | 1584 (10)  | 1587 (10)  | 1659 (10)  | 1670 (10)  | 1673 (10)  |
|                     | 1684 (10)  | 1687 (10)  | 12* (12)   | 62 (12)    | 157 (12)   | 207 (12)   | 207 (12)   | 228 (12)   |

|                            |            |            |            |            |            |            |            |            |
|----------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                            | 281 (12)   | 326 (12)   | 450 (12)   | 455 (12)   | 461 (12)   | 523 (12)   | 542 (12)   | 542 (12)   |
|                            | 559 (12)   | 576 (12)   | 1027 (13)  | 1100 (13)  | 1665 (13)  | 2557 (13)  | 2895 (13)  | 2931 (13)  |
|                            | 2935 (13)  | 2942 (13)  | 274 (14)   | 331 (14)   | 379 (14)   | 480 (14)   | 494 (14)   | 494 (14)   |
|                            | 729 (14)   | 742 (14)   | 1117 (14)  | 130 (15)   | 131 (15)   | 132 (15)   | 190 (15)   | 9 (16)     |
|                            | 29 (16)    | 30 (16)    | 31 (16)    | 41 (16)    | 44 (16)    | 45 (16)    | 71 (16)    | 142 (16)   |
|                            | 351 (16)   | 368 (16)   | 428 (16)   | 437 (16)   | 438 (16)   | 452 (16)   | 1090 (16)  | 1230 (16)  |
|                            | 1256 (16)  | 2028 (16)  | 2938 (16)  | 3082 (16)  | 3085 (16)  | 581 (17)   | 49 (18)    | 50 (18)    |
|                            | 64 (18)    | 97 (18)    | 136 (18)   | 1237 (18)  | 1872 (18)  | 80 (23)    | 157 (23)   | 237 (25)   |
|                            | 258 (25)   | 332 (25)   | 156 (26)   | 274 (26)   | 764 (26)   | 885 (26)   | 888 (26)   | 154 (27)   |
|                            | 264 (27)   | 273 (27)   | 276 (27)   | 327 (27)   | 341 (27)   | 357 (27)   | 28 (28)    | 33 (28)    |
|                            | 56 (28)    | 73 (28)    | 83 (28)    | 92 (28)    | 105 (28)   | 136 (28)   | 140 (28)   | 148 (28)   |
|                            | 154 (28)   | 227 (28)   | 302 (28)   | 411 (28)   | 414 (28)   | 444 (28)   | 459 (28)   | 459 (28)   |
|                            | 462 (28)   | 573 (28)   | 618 (28)   | 708 (28)   | 799 (28)   | 803 (28)   | 850 (28)   | 902 (28)   |
|                            | 935 (28)   | 939 (28)   | 939 (28)   | 948 (28)   | 958 (28)   | 964 (28)   | 964 (28)   | 970 (28)   |
|                            | 1059 (28)  | 1067 (28)  | 1114 (28)  | 168 (29)   | 43 (30)    | 325 (30)   | 376 (30)   | 379 (30)   |
|                            | 380 (30)   | 421 (30)   | 119 (31)   | 156 (31)   | 99 (33)    | 123 (33)   | 159 (33)   | 196 (33)   |
|                            | 240 (33)   | 43 (32)    | 85 (32)    | 112 (32)   | 189 (32)   | 198 (32)   | 201 (32)   | 107 (36)   |
|                            | 485 (36)   | 486 (36)   | 523 (36)   | 228 (37)   | 339 (37)   | 1374 (37)  | 1799 (37)  | 98 (38)    |
|                            | 99 (38)    | 259 (38)   | 290 (38)   | 325 (38)   | 380 (38)   | 390 (38)   | 393 (38)   | 565 (38)   |
|                            | 568 (38)   | 36 (39)    | 57 (39)    | 133 (39)   | 150 (39)   | 486 (40)   | 513 (40)   | 1049 (40)  |
|                            | 1076 (40)  | 1081 (40)  | 1138 (40)  | 1439 (40)  | 1466 (40)  | 1507 (40)  | 1508 (40)  | 1518 (40)  |
|                            | 1600 (40)  | 1608 (40)  | 16 (41)    | 81 (41)    | 96 (41)    | 98 (41)    | 124 (41)   | 215 (41)   |
|                            | 439 (41)   | 440 (41)   | 451 (41)   | 469 (41)   | 701 (41)   | 747 (41)   | 748 (41)   | 788 (41)   |
|                            | 815 (41)   |            |            |            |            |            |            |            |
| HandleActivateEvent        | 723* (13)  | 643 (16)   | 1303* (16) | 1321- (16) |            |            |            |            |
| HandleAlienEvent           | 708* (13)  | 659 (16)   | 1328* (16) | 1345- (16) |            |            |            |            |
| HandleCR                   | 104* (5)   | 500 (6)    | 570* (6)   | 581- (6)   |            |            |            |            |
| HandleCursor               | 1630* (13) | 3046 (16)  | 1263* (18) | 1277 (18)  | 1290- (18) |            |            |            |
| HandleDiskEvent            | 728* (13)  | 652 (16)   | 1352* (16) | 1371- (16) |            |            |            |            |
| HandleEvent                | 716* (13)  | 1379* (16) | 2362 (16)  | 3368 (16)  | 703 (19)   | 97 (37)    |            |            |
| HandleFailure              | 265* (6)   | 275 (6)    | 308* (6)   | 322 (6)    | 1129* (6)  | 1141 (6)   | 1399* (6)  | 1411 (6)   |
|                            | 1441* (6)  | 1457 (6)   | 1622* (6)  | 1634 (6)   | 1664* (6)  | 1680 (6)   | 1843* (6)  | 1854 (6)   |
|                            | 1878* (6)  | 1890 (6)   | 2035* (6)  | 2050 (6)   | 2089* (6)  | 2106 (6)   | 2493* (6)  | 2504 (6)   |
|                            | 2678* (6)  | 2691 (6)   | 1389* (16) | 1414 (16)  | 185* (18)  | 206 (18)   | 431* (18)  | 442 (18)   |
|                            | 27* (19)   | 61 (19)    | 96* (19)   | 115 (19)   | 214* (20)  | 220 (20)   | 631* (21)  | 640 (21)   |
|                            | 1213* (21) | 1224 (21)  | 1249* (21) | 1258 (21)  |            |            |            |            |
| HandleFinderRequest        | 822* (13)  | 1430* (16) | 2700 (16)  |            |            |            |            |            |
| HandleIsEligible           | 56* (28)   | 253 (28)   | 629 (28)   | 645 (28)   | 690 (28)   | 708* (28)  | 711- (28)  | 713- (28)  |
|                            | 1084 (28)  |            |            |            |            |            |            |            |
| HandleKeyDownEvent         | 732* (13)  | 649 (16)   | 1524* (16) | 1537- (16) | 1539- (16) |            |            |            |
| HandleMouseDown            | 737* (13)  | 1621* (13) | 1987* (13) | 640 (16)   | 1547* (16) | 1561- (16) | 1579 (16)  | 1588 (16)  |
|                            | 1601- (16) | 1614 (16)  | 1617- (16) | 1297* (18) | 1315 (18)  | 1322- (18) | 1330- (18) | 1334- (18) |
|                            | 684* (19)  | 709- (19)  | 709 (19)   | 711- (19)  |            |            |            |            |
| HandleMouseUp              | 744* (13)  | 637 (16)   | 1644* (16) | 1649- (16) |            |            |            |            |
| Handler                    | 189* (7)   | 19* (8)    |            |            |            |            |            |            |
| handler                    | 221* (7)   | 284* (8)   | 286 (8)    |            |            |            |            |            |
| HandleShiftKeyDown         | 681* (9)   | 2869* (10) | 2993 (10)  |            |            |            |            |            |
| HandlesPrintingCommands    | 656* (13)  | 1190* (13) | 287* (15)  | 290- (15)  | 290 (15)   | 292- (15)  | 401 (17)   | 605* (17)  |
|                            | 607- (17)  |            |            |            |            |            |            |            |
| HandleSystemEvent          | 748* (13)  | 656 (16)   | 1656* (16) | 1696- (16) |            |            |            |            |
| HandleTab                  | 106* (5)   | 498 (6)    | 588* (6)   | 604- (6)   |            |            |            |            |
| handleToText               | 398* (40)  | 424- (40)  | 430 (40)   | 486 (40)   | 499 (40)   | 503 (40)   | 508 (40)   | 510 (40)   |
|                            | 513 (40)   |            |            |            |            |            |            |            |
| HandleUpdateEvent          | 754* (13)  | 646 (16)   | 1703* (16) | 1722- (16) |            |            |            |            |
| HandleZone                 | 2099 (16)  | 1072 (28)  |            |            |            |            |            |            |
| HandToHand                 | 1275 (6)   | 2787 (6)   | 462 (28)   | 202 (33)   |            |            |            |            |
| handyRect                  | 353* (37)  | 376- (37)  | 378 (37)   | 382- (37)  | 383 (37)   |            |            |            |
| hasColorQD                 | 145 (4)    | 78 (6)     | 116 (6)    | 167 (6)    | 1412 (6)   | 1458 (6)   | 1635 (6)   | 1681 (6)   |
|                            | 2267 (6)   | 583 (14)   | 702 (14)   | 24 (16)    | 27 (16)    | 1275 (16)  | 1842 (18)  | 1891 (18)  |
|                            | 119 (19)   | 177* (25)  | 299 (26)   | 787 (26)   | 371 (30)   | 533 (41)   |            |            |
| hasDesktopBus              | 190* (25)  | 113- (26)  |            |            |            |            |            |            |
| hasFPU                     | 176* (25)  |            |            |            |            |            |            |            |
| hasGoAway                  | 1868* (13) | 121 (19)   | 124 (19)   | 129 (19)   | 188- (19)  |            |            |            |
| hasHFS                     | 701 (14)   | 532 (17)   | 688 (17)   | 183* (25)  | 117- (26)  | 119- (26)  | 208 (26)   | 612 (26)   |
|                            | 647 (26)   |            |            |            |            |            |            |            |
| hasHierarchicalMenus       | 2044 (6)   | 2098 (6)   | 131 (16)   | 184* (25)  | 120- (26)  | 218 (30)   | 453 (30)   |            |
| hasROM128K                 | 2652 (6)   | 3192 (6)   | 404 (14)   | 422 (14)   | 609 (14)   | 682 (14)   | 700 (14)   | 1264 (16)  |
|                            | 593 (19)   | 182* (25)  | 63 (26)    | 115- (26)  | 116 (26)   | 120 (26)   | 121 (26)   | 123 (26)   |
|                            | 124 (26)   | 278 (26)   | 364 (26)   | 635 (26)   | 768 (26)   | 1152 (26)  | 514 (28)   | 76 (35)    |
|                            | 105 (35)   | 204 (40)   | 1337 (40)  |            |            |            |            |            |
| hasScriptManager           | 3188 (6)   | 613 (14)   | 678 (14)   | 1077 (14)  | 185* (25)  | 64 (26)    | 121- (26)  | 365 (26)   |
|                            | 373 (26)   | 249 (40)   | 490 (40)   | 1185 (40)  | 1392 (40)  | 719 (41)   | 793 (41)   |            |
| hasSCSI                    | 189* (25)  | 114- (26)  |            |            |            |            |            |            |
| hasSoundManager            | 187* (25)  | 123- (26)  |            |            |            |            |            |            |
| hasStyleTextEdit           | 186* (25)  | 122- (26)  | 433 (40)   |            |            |            |            |            |
| hasStyleTextedit           | 41 (40)    | 103 (40)   |            |            |            |            |            |            |
| hasWaitNextEvent           | 703 (14)   | 1137 (16)  | 188* (25)  | 124- (26)  |            |            |            |            |
| haveCursorRgn              | 3009* (16) | 3039- (16) | 3067- (16) | 3072 (16)  |            |            |            |            |
| haveEvent                  | 1123* (16) | 1160- (16) | 1175- (16) | 1186- (16) | 1191- (16) | 1196- (16) | 1200 (16)  |            |
| haveHPrint                 | 1795* (37) | 1802- (37) | 1809- (37) | 1813 (37)  |            |            |            |            |
| HaveScrollBar              | 1828* (13) | 349* (20)  | 1232 (21)  | 1305 (21)  |            |            |            |            |
| HBoolArray                 | 15* (28)   | 23 (28)    | 464 (28)   |            |            |            |            |            |
| hCentered                  | 1888* (13) |            |            |            |            |            |            |            |
| hConstrain                 | 1752* (13) | 64 (20)    | 100- (20)  |            |            |            |            |            |
| hDeltaOrg                  | 1586* (13) | 579* (18)  | 589 (18)   |            |            |            |            |            |
| HdlCharFailed              | 824* (41)  | 836 (41)   |            |            |            |            |            |            |
| HdlClipFailed              | 357* (41)  | 381 (41)   |            |            |            |            |            |            |
| HdlDefaultError            | 1710* (37) | 1722 (37)  |            |            |            |            |            |            |
| HdlDoCreateViews           | 117* (15)  | 134 (15)   |            |            |            |            |            |            |
| HdlDoIt                    | 2252* (16) | 2287 (16)  |            |            |            |            |            |            |
| HdlEach                    | 259* (12)  | 269 (12)   |            |            |            |            |            |            |
| HdlFailFailed              | 1324* (17) | 1333 (17)  |            |            |            |            |            |            |
| HdlFailure                 | 1237* (16) | 1252 (16)  | 165* (23)  | 209 (23)   |            |            |            |            |
| HdlGivePasteFailed         | 1089* (40) | 1109 (40)  |            |            |            |            |            |            |
| HdlIDocument               | 19* (17)   | 79 (17)    |            |            |            |            |            |            |
| HdlIEntry                  | 21* (2)    | 29 (2)     |            |            |            |            |            |            |
| HdlInitFailed              | 253* (37)  | 263 (37)   | 2099* (37) | 2109 (37)  | 22* (41)   | 51 (41)    | 639* (41)  | 647 (41)   |
| HdlMakeFailed              | 1209* (40) | 1231 (40)  |            |            |            |            |            |            |
| HdlMakeViewForAlienClipbds | 2495* (16) | 2506 (16)  |            |            |            |            |            |            |

[illegible]

|                    |           |           |           |           |           |           |           |           |  |
|--------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| horzMax            | 1748*(13) | 60 (20)   | 96=(20)   |           |           |           |           |           |  |
| horzUnits          | 1821*(13) | 637*(20)  | 641 (20)  |           |           |           |           |           |  |
| HParamBlockRec     | 339 (16)  | 243 (17)  | 301 (17)  | 1043 (17) | 406 (25)  | 134 (26)  | 173 (26)  | 233 (26)  |  |
|                    | 583 (26)  |           |           |           |           |           |           |           |  |
| HParmBlkPtr        | 319 (16)  | 393 (25)  | 194 (26)  |           |           |           |           |           |  |
| hPB                | 1043*(17) | 1111 (17) | 1117 (17) | 1121 (17) | 1124 (17) | 1144 (17) | 1149 (17) | 1150 (17) |  |
|                    | 1181 (17) | 1184 (17) | 1184 (17) | 134*(26)  | 137 (26)  | 144 (26)  | 147 (26)  |           |  |
| HPurge             | 1569 ( 6) | 1789 ( 6) | 2447 ( 6) | 520 (14)  | 678 (28)  | 827 (28)  |           |           |  |
| HRunArray          | 84*( 9)   | 140 ( 9)  | 141 ( 9)  | 45 (10)   | 73 (10)   | 77 (10)   | 142 (10)  | 179 (10)  |  |
|                    | 183 (10)  |           |           |           |           |           |           |           |  |
| hScrollUnits       | 1750*(13) | 64 (20)   | 98=(20)   |           |           |           |           |           |  |
| HSegHeader         | 893*(28)  |           |           |           |           |           |           |           |  |
| HSetState          | 772 (26)  |           |           |           |           |           |           |           |  |
| HSFTypeList        | 410*(16)  | 439 (16)  |           |           |           |           |           |           |  |
| HString\$          | 141*(25)  |           |           |           |           |           |           |           |  |
| hStrip             | 252*(36)  | 2031*(37) | 2037 (37) |           |           |           |           |           |  |
| hTE                | 37*(38)   | 41*(38)   | 44*(38)   | 47*(38)   | 605*(38)  | 163*(39)  | 165 (39)  |           |  |
| HTemplate          | 1224*(16) | 1228 (16) | 1254 (16) |           |           |           |           |           |  |
| hText              | 96 (23)   | 97=(23)   | 104=(23)  | 36*(39)   | 45 (39)   | 57*(39)   | 67 (39)   | 73=(39)   |  |
|                    | 73 (39)   | 77 (39)   | 56 (40)   | 111 (40)  | 183=(40)  | 186 (40)  | 307 (40)  | 424 (40)  |  |
|                    | 1531 (40) | 1533=(40) | 1537 (40) | 59 (41)   | 834 (41)  |           |           |           |  |
| HTypeList          | 551*(13)  | 872 (13)  | 337 (16)  | 346 (16)  | 419 (16)  | 424 (16)  | 2928 (16) |           |  |
| HUnlock            | 1289 ( 6) | 2821 ( 6) | 1123 (14) | 190 (15)  | 31 (16)   | 44 (16)   | 3085 (16) | 67 (18)   |  |
|                    | 121 (18)  | 186 (28)  | 379 (30)  | 201 (32)  | 513 (40)  | 1093 (40) | 1152 (40) | 146 (41)  |  |
|                    | 235 (41)  |           |           |           |           |           |           |           |  |
| HWCfgFlags         | 79*(26)   | 113 (26)  | 114 (26)  |           |           |           |           |           |  |
| HWStateData        | 870*(19)  | 880 (19)  |           |           |           |           |           |           |  |
| HX                 | 42*(39)   | 45 (39)   |           |           |           |           |           |           |  |
| hysteresis         | 470*( 5)  | 589*( 5)  | 2240*( 6) | 3062*( 6) | 3074 ( 6) | 224*( 9)  | 620*(10)  | 741*(13)  |  |
|                    | 1622*(13) | 1626*(13) | 1989*(13) | 2100*(13) | 2176*(13) | 2249*(13) | 2305*(13) | 1558*(16) |  |
|                    | 1613*(16) | 1615 (16) | 1617 (16) | 3098*(16) | 3284 (16) | 3285 (16) | 803*(18)  | 1299*(18) |  |
|                    | 1316 (18) | 1333 (18) | 686*(19)  | 709 (19)  | 303*(21)  | 677*(21)  | 1049*(21) | 1342*(21) |  |
|                    | 1377 (21) | 196*(38)  | 884*(40)  |           |           |           |           |           |  |
| -I-                |           |           |           |           |           |           |           |           |  |
| i                  | 438*(10)  | 466=(10)  | 467 (10)  | 692*(10)  | 728=(10)  | 733 (10)  | 735 (10)  | 747=(10)  |  |
|                    | 752 (10)  | 754 (10)  | 771*(10)  | 796=(10)  | 801 (10)  | 804 (10)  | 844*(10)  | 879=(10)  |  |
|                    | 941*(10)  | 976=(10)  | 1084*(10) | 1110=(10) | 1113 (10) | 1792*(10) | 1820=(10) | 1821 (10) |  |
|                    | 2074*(10) | 2093=(10) | 2099 (10) | 2099 (10) | 2100 (10) | 2101=(10) | 2101 (10) | 2103 (10) |  |
|                    | 2104 (10) | 2116=(10) | 2122 (10) | 2122 (10) | 2123 (10) | 2124=(10) | 2124 (10) | 2126 (10) |  |
|                    | 2127 (10) | 2560*(10) | 2563=(10) | 2564 (10) | 383*(12)  | 389=(12)  | 396=(12)  | 396 (12)  |  |
|                    | 399 (12)  | 624*(12)  | 635=(12)  | 637 (12)  | 639 (12)  | 256*(14)  | 259=(14)  | 260 (14)  |  |
|                    | 262 (14)  | 332*(14)  | 734*(14)  | 747=(14)  | 107*(15)  | 141=(15)  | 186 (15)  | 73*(16)   |  |
|                    | 136=(16)  | 138 (16)  | 338*(16)  | 353=(16)  | 356 (16)  | 1434*(16) | 1492=(16) | 1496 (16) |  |
|                    | 1500 (16) | 482*(26)  | 487=(26)  | 489 (26)  | 693*(26)  | 698 (26)  | 700 (26)  | 905*(26)  |  |
|                    | 1005=(26) | 1006 (26) | 1006 (26) | 1011=(26) | 1013 (26) | 1130*(26) | 1133=(26) | 1134 (26) |  |
|                    | 1135 (26) | 1135 (26) | 104*(28)  | 113=(28)  | 115 (28)  | 155*(28)  | 165=(28)  | 170 (28)  |  |
|                    | 175=(28)  | 175 (28)  | 409*(28)  | 441=(28)  | 444 (28)  | 449 (28)  | 451 (28)  | 452 (28)  |  |
|                    | 455=(28)  | 455 (28)  | 465=(28)  | 466 (28)  | 484=(28)  | 486 (28)  | 491=(28)  | 493 (28)  |  |
|                    | 570*(28)  | 797*(28)  | 813=(28)  | 961*(28)  | 964=(28)  | 967 (28)  | 971=(28)  | 971 (28)  |  |
|                    | 1113*(28) | 1131=(28) | 1132 (28) | 1134 (28) | 1138 (28) | 61*(30)   | 72=(30)   | 73 (30)   |  |
|                    | 102*(30)  | 190*(30)  | 202=(30)  | 204 (30)  | 226=(30)  | 316*(30)  | 329=(30)  | 331 (30)  |  |
|                    | 336 (30)  | 65*(33)   | 100=(33)  | 41*(32)   | 66=(32)   | 69 (32)   | 74=(32)   | 75 (32)   |  |
|                    | 77=(32)   | 77 (32)   | 106*(32)  | 112=(32)  | 113 (32)  | 115 (32)  | 121=(32)  | 121 (32)  |  |
|                    | 777*(37)  | 798*(37)  | 800 (37)  | 803 (37)  | 805 (37)  | 809 (37)  | 813 (37)  | 817 (37)  |  |
|                    | 974*(37)  | 996=(37)  | 58*(39)   | 75=(39)   | 77 (39)   | 78 (39)   | 104*(39)  | 126=(39)  |  |
|                    | 127 (39)  | 129 (39)  | 143=(39)  | 144 (39)  | 146 (39)  | 911*(41)  | 921=(41)  | 922 (41)  |  |
|                    | 923 (41)  |           |           |           |           |           |           |           |  |
| i0                 | 1690*(37) |           |           |           |           |           |           |           |  |
| IApplication       | 687*(13)  | 58*(16)   |           |           |           |           |           |           |  |
| IAssociation       | 63*( 1)   | 91*( 2)   | 278 ( 6)  | 325 ( 6)  | 692 (14)  |           |           |           |  |
| iBeamCursor        | 920 (40)  |           |           |           |           |           |           |           |  |
| IButton            | 141*( 5)  | 780*( 6)  |           |           |           |           |           |           |  |
| ICellSelectCommand | 677*( 9)  | 642 (10)  | 2853*(10) |           |           |           |           |           |  |
| ICheckBox          | 171*( 5)  | 861*( 6)  |           |           |           |           |           |           |  |
| ICluster           | 254*( 5)  | 1119*( 6) |           |           |           |           |           |           |  |
| IColSelectCommand  | 737*( 9)  | 660 (10)  | 3112*(10) |           |           |           |           |           |  |
| ICommand           | 2785 (10) | 2461*(13) | 38 (21)   | 12*(22)   | 2108 (37) | 50 (41)   |           |           |  |
| IconTemplate       | 293 ( 5)  | 294*( 5)  | 1471 ( 6) | 1489 ( 6) |           |           |           |           |  |
| IconTemplatePtr    | 293*( 5)  | 1450 ( 6) | 1484 ( 6) | 1489 ( 6) |           |           |           |           |  |
| IControl           | 1136 ( 6) | 1406 ( 6) | 1629 ( 6) | 1850 ( 6) | 2042 ( 6) | 2685 ( 6) | 2073*(13) | 97*(21)   |  |
|                    | 540 (21)  |           |           |           |           |           |           |           |  |
| IControlTracker    | 2532*(13) | 36*(21)   | 309 (21)  |           |           |           |           |           |  |
| iCopies            | 1153 (37) |           |           |           |           |           |           |           |  |
| icPtr              | 1484*( 6) | 1489=( 6) | 1491 ( 6) |           |           |           |           |           |  |
| ICtlMgr            | 784 ( 6)  | 865 ( 6)  | 993 ( 6)  | 2153*(13) | 532*(21)  | 945 (21)  |           |           |  |
| id                 | 757*(28)  | 776 (28)  |           |           |           |           |           |           |  |
| IdentifyPoint      | 319*( 9)  | 633 (10)  | 1448*(10) | 1466=(10) | 1470=(10) | 1476=(10) | 1481=(10) | 1490 (10) |  |
|                    | 1491=(10) | 1493=(10) | 1498 (10) | 1499=(10) | 1501=(10) |           |           |           |  |
| IdentifySoftware   | 601*(13)  | 1061*(13) | 301*(15)  | 304 (15)  | 776 (16)  | 1731*(16) | 462*(36)  | 1035*(37) |  |
|                    | 353*(38)  | 1620*(40) | 1623 (40) |           |           |           |           |           |  |
| IDeskScrapView     | 2561*(13) | 148 (16)  | 15*(23)   |           |           |           |           |           |  |
| iDev               | 549 (37)  |           |           |           |           |           |           |           |  |
| IDialogView        | 66*( 5)   | 253*( 6)  |           |           |           |           |           |           |  |
| Idle               | 933*(13)  | 1156 (16) | 1745*(16) | 2359 (16) | 2371 (16) |           |           |           |  |
| IdleBegin          | 517*(13)  | 1154 (16) | 1156 (16) | 1189 (16) | 1763 (16) | 1776 (16) | 1904 (16) | 1906 (16) |  |
|                    | 2357 (16) | 2360 (16) |           |           |           |           |           |           |  |
| IdleContinue       | 517*(13)  | 1157 (16) | 2372 (16) |           |           |           |           |           |  |
| IdleEnd            | 517*(13)  | 574 (14)  | 1799 (16) | 2359 (16) |           |           |           |           |  |
| IdlePhase          | 582 ( 5)  | 3007 ( 6) | 517*(13)  | 613 (13)  | 933 (13)  | 2683 (13) | 224 (15)  | 1745 (16) |  |
|                    | 211 (38)  | 675 (40)  |           |           |           |           |           |           |  |
| IDocument          | 1140*(13) | 10*(17)   |           |           |           |           |           |           |  |
| IDType             | 52 ( 5)   | 53 ( 5)   | 57 ( 5)   | 58 ( 5)   | 63 ( 5)   | 69 ( 5)   | 84 ( 5)   | 90 ( 5)   |  |
|                    | 114 ( 5)  | 118 ( 5)  | 279 ( 5)  | 256 ( 6)  | 392 ( 6)  | 435 ( 6)  | 643 ( 6)  | 698 ( 6)  |  |
|                    | 1324 ( 6) | 1437 (13) | 1438 (13) | 1446 (13) | 1468 (13) | 1538 (13) | 1882 (13) | 1910 (13) |  |
|                    | 2769 (13) | 2812 (13) | 2991 (13) | 2994 (13) | 253 (14)  | 1171 (14) | 112 (15)  | 947 (18)  |  |
|                    | 143*(25)  | 488 (25)  | 490 (25)  | 1316 (26) | 1322 (26) | 1330 (26) |           |           |  |
| IDUObject          | 1734 (16) |           |           |           |           |           |           |           |  |
| IDUObject          | 321*(32)  |           |           |           |           |           |           |           |  |

[illegible]

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| InitUMemory       | 345 (14)  | 204*(27)  | 723*(28)  | 735=(28)  |           |           |           |           |  |
| InitUMenuSetup    | 544 (14)  | 227*(29)  | 305*(30)  |           |           |           |           |           |  |
| InitUPatch        | 411 (14)  | 73*(34)   | 50*(35)   |           |           |           |           |           |  |
| InitUTEView       | 602*(38)  | 21*(39)   |           |           |           |           |           |           |  |
| initVal           | 63*(33)   | 102 (33)  |           |           |           |           |           |           |  |
| InitWindows       | 386 (14)  |           |           |           |           |           |           |           |  |
| inMenuBar         | 1575 (16) | 1576 (16) | 1598 (16) | 2650 (16) |           |           |           |           |  |
| InModalMenuState  | 780*(13)  | 673 (16)  | 1575 (16) | 1823*(16) | 1826=(16) |           |           |           |  |
| InModalState      | 777*(13)  | 1576 (16) | 1813*(16) | 1816=(16) | 2860 (16) |           |           |           |  |
| inner             | 320*(25)  | 727*(26)  | 729 (26)  |           |           |           |           |           |  |
| inPageDown        | 566 (20)  | 1056 (21) |           |           |           |           |           |           |  |
| inPageUp          | 565 (20)  | 582 (20)  | 21 (21)   | 1056 (21) |           |           |           |           |  |
| inPort            | 83 (39)   | 84 (39)   | 86 (39)   | 221=(40)  | 223=(40)  | 990 (40)  | 992=(40)  | 1025=(40) |  |
| InRow             | 1493 (10) | 1501 (10) |           |           |           |           |           |           |  |
| inRow             | 76*(9)    | 647 (10)  |           |           |           |           |           |           |  |
| InsColBefore      | 325*(9)   | 1515*(10) | 1717 (10) | 1741 (10) | 1897 (10) |           |           |           |  |
| InsColFirst       | 331*(9)   | 127 (10)  | 225 (10)  | 1739*(10) |           |           |           |           |  |
| InsColLast        | 337*(9)   | 1715*(10) |           |           |           |           |           |           |  |
| Insert            | 229 (2)   | 272*(11)  | 715*(12)  | 1252 (14) |           |           |           |           |  |
| InsertBefore      | 184*(11)  | 418*(12)  | 476 (12)  | 487 (12)  | 718 (12)  |           |           |           |  |
| InsertEntry       | 79*(1)    | 190*(2)   | 635 (6)   | 1159 (14) |           |           |           |           |  |
| InsertFirst       | 194*(11)  | 472*(12)  | 1466 (18) |           |           |           |           |           |  |
| InsertLast        | 202*(11)  | 483*(12)  | 254 (16)  | 263 (16)  | 127 (17)  | 148 (17)  | 354 (18)  | 1481 (18) |  |
|                   | 1231 (21) |           |           |           |           |           |           |           |  |
| InsertMenu        | 121 (16)  | 140 (16)  | 366 (30)  |           |           |           |           |           |  |
| Inset             | 2937 (6)  | 2110*(13) | 378*(21)  |           |           |           |           |           |  |
| InsetRect         | 1268 (6)  | 2211 (6)  | 2227 (6)  | 2405 (6)  | 525 (14)  |           |           |           |  |
| InsetRgn          | 588 (19)  |           |           |           |           |           |           |           |  |
| InsetVRect        | 3345 (6)  | 1459 (10) | 54*(42)   |           |           |           |           |           |  |
| InsItemBefore     | 591*(9)   | 2640*(10) | 2655 (10) | 2668 (10) |           |           |           |           |  |
| InsItemFirst      | 592*(9)   | 2653*(10) |           |           |           |           |           |           |  |
| InsItemLast       | 593*(9)   | 2666*(10) |           |           |           |           |           |           |  |
| Inspect           | 86*(31)   | 199 (32)  | 303*(32)  |           |           |           |           |           |  |
| InspectField      | 152*(32)  | 258 (32)  | 310 (32)  |           |           |           |           |           |  |
| InspectObject     | 495 (14)  | 119*(31)  | 189*(32)  |           |           |           |           |           |  |
| InspectorName     | 131=(24)  |           |           |           |           |           |           |           |  |
| inspectorName     | 246*(11)  | 651*(12)  | 655=(12)  | 695*(13)  | 1157*(13) | 1735*(13) | 2035*(13) | 2344*(13) |  |
|                   | 2578*(13) | 1078*(16) | 1082=(16) | 503*(17)  | 506=(17)  | 1103*(18) | 1113 (18) | 1114=(18) |  |
|                   | 1114 (18) | 649*(19)  | 653 (19)  | 282*(23)  | 286=(23)  | 288=(23)  | 120*(24)  | 127=(24)  |  |
|                   | 94*(31)   | 264*(33)  |           |           |           |           |           |           |  |
| InsRowBefore      | 326*(9)   | 1615*(10) | 1729 (10) | 1753 (10) | 1923 (10) | 2642 (10) |           |           |  |
| InsRowFirst       | 332*(9)   | 125 (10)  | 223 (10)  | 1751*(10) |           |           |           |           |  |
| InsRowLast        | 338*(9)   | 1727*(10) |           |           |           |           |           |           |  |
| InstallCohandler  | 794*(13)  | 1834*(16) |           |           |           |           |           |           |  |
| InstallColor      | 2116*(13) | 393*(21)  |           |           |           |           |           |           |  |
| InstallDocument   | 2026*(13) | 71 (19)   | 147 (19)  | 719*(19)  |           |           |           |           |  |
| InstallEditText   | 679*(5)   | 2576*(6)  | 3338 (6)  |           |           |           |           |           |  |
| InstallManyStyles | 520*(38)  | 551*(41)  | 588 (41)  |           |           |           |           |           |  |
| InstallMargins    | 365*(36)  | 1046*(37) | 1836 (37) |           |           |           |           |           |  |
| InstallNewText    | 430*(38)  | 121*(41)  | 253 (41)  |           |           |           |           |           |  |
| InstallOneStyle   | 517*(38)  | 541*(41)  | 570 (41)  | 586 (41)  |           |           |           |           |  |
| InstallSelection  | 602*(5)   | 681*(5)   | 2616*(6)  | 2628 (6)  | 3312*(6)  | 3322 (6)  | 3367 (6)  | 658*(13)  |  |
|                   | 313*(15)  | 2790 (16) | 2791 (16) | 205*(38)  | 1178*(40) |           |           |           |  |
| InstallTextStyle  | 2118*(13) | 404*(21)  |           |           |           |           |           |           |  |
| InstallTitle      | 1385*(17) | 1391 (17) |           |           |           |           |           |           |  |
| InstallTracker    | 3173*(16) | 3236 (16) | 3239 (16) | 3244 (16) |           |           |           |           |  |
| inSysWindow       | 1604 (16) | 2652 (16) |           |           |           |           |           |           |  |
| intDrawHook       | 34*(38)   |           |           |           |           |           |           |           |  |
| INTEGER           | 40 (1)    | 51 (1)    | 88 (1)    | 57 (2)    | 73 (2)    | 144 (2)   | 275 (2)   | 326 (2)   |  |
|                   | 357 (2)   | 79 (3)    | 15 (4)    | 60 (4)    | 88 (4)    | 201 (4)   | 95 (5)    | 108 (5)   |  |
|                   | 124 (5)   | 186 (5)   | 226 (5)   | 249 (5)   | 250 (5)   | 256 (5)   | 273 (5)   | 285 (5)   |  |
|                   | 296 (5)   | 302 (5)   | 308 (5)   | 331 (5)   | 342 (5)   | 348 (5)   | 354 (5)   | 378 (5)   |  |
|                   | 388 (5)   | 393 (5)   | 399 (5)   | 422 (5)   | 432 (5)   | 433 (5)   | 434 (5)   | 439 (5)   |  |
|                   | 440 (5)   | 442 (5)   | 443 (5)   | 448 (5)   | 478 (5)   | 480 (5)   | 484 (5)   | 486 (5)   |  |
|                   | 491 (5)   | 501 (5)   | 507 (5)   | 508 (5)   | 509 (5)   | 515 (5)   | 537 (5)   | 541 (5)   |  |
|                   | 547 (5)   | 557 (5)   | 564 (5)   | 569 (5)   | 584 (5)   | 586 (5)   | 600 (5)   | 610 (5)   |  |
|                   | 612 (5)   | 624 (5)   | 666 (5)   | 689 (5)   | 113 (6)   | 145 (6)   | 447 (6)   | 493 (6)   |  |
|                   | 756 (6)   | 936 (6)   | 1063 (6)  | 1121 (6)  | 1222 (6)  | 1247 (6)  | 1248 (6)  | 1249 (6)  |  |
|                   | 1369 (6)  | 1392 (6)  | 1592 (6)  | 1615 (6)  | 1812 (6)  | 1835 (6)  | 2004 (6)  | 2026 (6)  |  |
|                   | 2189 (6)  | 2190 (6)  | 2191 (6)  | 2242 (6)  | 2367 (6)  | 2368 (6)  | 2369 (6)  | 2419 (6)  |  |
|                   | 2428 (6)  | 2460 (6)  | 2484 (6)  | 2529 (6)  | 2563 (6)  | 2670 (6)  | 2846 (6)  | 2866 (6)  |  |
|                   | 2904 (6)  | 2926 (6)  | 3018 (6)  | 3136 (6)  | 3155 (6)  | 3175 (6)  | 3256 (6)  | 3267 (6)  |  |
|                   | 3515 (6)  | 3542 (6)  | 153 (7)   | 183 (7)   | 189 (7)   | 193 (7)   | 206 (7)   | 218 (7)   |  |
|                   | 19 (8)    | 26 (8)    | 74 (8)    | 86 (8)    | 141 (8)   | 149 (8)   | 226 (8)   | 78 (9)    |  |
|                   | 79 (9)    | 93 (9)    | 94 (9)    | 95 (9)    | 96 (9)    | 97 (9)    | 98 (9)    | 112 (9)   |  |
|                   | 138 (9)   | 139 (9)   | 144 (9)   | 145 (9)   | 155 (9)   | 156 (9)   | 157 (9)   | 158 (9)   |  |
|                   | 160 (9)   | 161 (9)   | 162 (9)   | 164 (9)   | 175 (9)   | 176 (9)   | 177 (9)   | 178 (9)   |  |
|                   | 181 (9)   | 182 (9)   | 235 (9)   | 236 (9)   | 269 (9)   | 269 (9)   | 270 (9)   | 270 (9)   |  |
|                   | 279 (9)   | 279 (9)   | 280 (9)   | 280 (9)   | 284 (9)   | 285 (9)   | 289 (9)   | 290 (9)   |  |
|                   | 306 (9)   | 306 (9)   | 307 (9)   | 309 (9)   | 309 (9)   | 310 (9)   | 315 (9)   | 315 (9)   |  |
|                   | 316 (9)   | 316 (9)   | 320 (9)   | 325 (9)   | 325 (9)   | 325 (9)   | 326 (9)   | 326 (9)   |  |
|                   | 326 (9)   | 331 (9)   | 331 (9)   | 332 (9)   | 332 (9)   | 337 (9)   | 337 (9)   | 338 (9)   |  |
|                   | 338 (9)   | 362 (9)   | 362 (9)   | 362 (9)   | 363 (9)   | 363 (9)   | 363 (9)   | 380 (9)   |  |
|                   | 400 (9)   | 419 (9)   | 420 (9)   | 431 (9)   | 432 (9)   | 433 (9)   | 436 (9)   | 437 (9)   |  |
|                   | 497 (9)   | 526 (9)   | 529 (9)   | 530 (9)   | 552 (9)   | 556 (9)   | 562 (9)   | 569 (9)   |  |
|                   | 569 (9)   | 570 (9)   | 571 (9)   | 575 (9)   | 576 (9)   | 579 (9)   | 582 (9)   | 582 (9)   |  |
|                   | 585 (9)   | 591 (9)   | 591 (9)   | 592 (9)   | 593 (9)   | 597 (9)   | 606 (9)   | 611 (9)   |  |
|                   | 611 (9)   | 612 (9)   | 616 (9)   | 625 (9)   | 662 (9)   | 700 (9)   | 712 (9)   | 715 (9)   |  |
|                   | 723 (9)   | 735 (9)   | 738 (9)   | 746 (9)   | 757 (9)   | 758 (9)   | 761 (9)   | 769 (9)   |  |
|                   | 33 (10)   | 34 (10)   | 35 (10)   | 36 (10)   | 39 (10)   | 40 (10)   | 329 (10)  | 339 (10)  |  |
|                   | 432 (10)  | 432 (10)  | 435 (10)  | 437 (10)  | 438 (10)  | 624 (10)  | 692 (10)  | 696 (10)  |  |
|                   | 697 (10)  | 769 (10)  | 770 (10)  | 771 (10)  | 773 (10)  | 839 (10)  | 839 (10)  | 843 (10)  |  |
|                   | 844 (10)  | 846 (10)  | 848 (10)  | 936 (10)  | 936 (10)  | 940 (10)  | 941 (10)  | 943 (10)  |  |
|                   | 945 (10)  | 1033 (10) | 1045 (10) | 1057 (10) | 1068 (10) | 1084 (10) | 1085 (10) | 1086 (10) |  |
|                   | 1087 (10) | 1088 (10) | 1163 (10) | 1163 (10) | 1164 (10) | 1166 (10) | 1167 (10) | 1168 (10) |  |
|                   | 1247 (10) | 1247 (10) | 1248 (10) | 1250 (10) | 1251 (10) | 1252 (10) | 1342 (10) | 1342 (10) |  |
|                   | 1344 (10) | 1346 (10) | 1373 (10) | 1373 (10) | 1375 (10) | 1377 (10) | 1449 (10) | 1515 (10) |  |
|                   | 1515 (10) | 1515 (10) | 1522 (10) | 1524 (10) | 1615 (10) | 1615 (10) | 1615 (10) | 1622 (10) |  |
|                   | 1624 (10) | 1715 (10) | 1715 (10) | 1727 (10) | 1727 (10) | 1739 (10) | 1739 (10) | 1751 (10) |  |

|                |           |           |            |           |            |           |           |
|----------------|-----------|-----------|------------|-----------|------------|-----------|-----------|
| 1751 (10)      | 1786 (10) | 1786 (10) | 1789 (10)  | 1791 (10) | 1792 (10)  | 1882 (10) | 1882 (10) |
| 1882 (10)      | 1908 (10) | 1908 (10) | 1908 (10)  | 2044 (10) | 2074 (10)  | 2140 (10) | 2203 (10) |
| 2204 (10)      | 2205 (10) | 2208 (10) | 2209 (10)  | 2273 (10) | 2406 (10)  | 2445 (10) | 2448 (10) |
| 2449 (10)      | 2485 (10) | 2507 (10) | 2520 (10)  | 2520 (10) | 2533 (10)  | 2546 (10) | 2557 (10) |
| 2558 (10)      | 2560 (10) | 2573 (10) | 2589 (10)  | 2589 (10) | 2600 (10)  | 2611 (10) | 2640 (10) |
| 2640 (10)      | 2653 (10) | 2666 (10) | 2677 (10)  | 2721 (10) | 2737 (10)  | 2737 (10) | 2738 (10) |
| 2749 (10)      | 2762 (10) | 2833 (10) | 3059 (10)  | 3080 (10) | 3094 (10)  | 3113 (10) | 3127 (10) |
| 3147 (10)      | 3162 (10) | 51 (11)   | 55 (11)    | 59 (11)   | 78 (11)    | 81 (11)   | 91 (11)   |
| 179 (11)       | 184 (11)  | 226 (11)  | 249 (11)   | 260 (11)  | 277 (11)   | 283 (11)  | 76 (12)   |
| 82 (12)        | 102 (12)  | 107 (12)  | 151 (12)   | 154 (12)  | 240 (12)   | 315 (12)  | 378 (12)  |
| 406 (12)       | 418 (12)  | 512 (12)  | 622 (12)   | 624 (12)  | 664 (12)   | 677 (12)  | 680 (12)  |
| 683 (12)       | 684 (12)  | 726 (12)  | 729 (12)   | 732 (12)  | 734 (12)   | 512 (13)  | 609 (13)  |
| 633 (13)       | 650 (13)  | 699 (13)  | 703 (13)   | 787 (13)  | 829 (13)   | 870 (13)  | 921 (13)  |
| 977 (13)       | 977 (13)  | 1106 (13) | 1125 (13)  | 1127 (13) | 1129 (13)  | 1131 (13) | 1161 (13) |
| 1179 (13)      | 1181 (13) | 1192 (13) | 1193 (13)  | 1194 (13) | 1232 (13)  | 1277 (13) | 1287 (13) |
| 1312 (13)      | 1318 (13) | 1334 (13) | 1348 (13)  | 1379 (13) | 1447* (13) | 1449 (13) | 1455 (13) |
| 1536 (13)      | 1556 (13) | 1586 (13) | 1698 (13)  | 1733 (13) | 1749 (13)  | 1750 (13) | 1835 (13) |
| 1857 (13)      | 1867 (13) | 1906 (13) | 1911 (13)  | 1912 (13) | 1969 (13)  | 2014 (13) | 2014 (13) |
| 2033 (13)      | 2055 (13) | 2062 (13) | 2110 (13)  | 2140 (13) | 2155 (13)  | 2171 (13) | 2180 (13) |
| 2182 (13)      | 2186 (13) | 2192 (13) | 2194 (13)  | 2198 (13) | 2200 (13)  | 2207 (13) | 2223 (13) |
| 2265 (13)      | 2272 (13) | 2309 (13) | 2316 (13)  | 2335 (13) | 2348 (13)  | 2375 (13) | 2404 (13) |
| 2409 (13)      | 2415 (13) | 2468 (13) | 2554 (13)  | 2582 (13) | 2610 (13)  | 2660 (13) | 2670 (13) |
| 2690 (13)      | 2708 (13) | 2715 (13) | 2717 (13)  | 2718 (13) | 2720 (13)  | 2730 (13) | 2756 (13) |
| 2771 (13)      | 2775 (13) | 2851 (13) | 2854 (13)  | 2860 (13) | 2865 (13)  | 2871 (13) | 2876 (13) |
| 2884 (13)      | 2910 (13) | 2910 (13) | 2918 (13)  | 2918 (13) | 2921 (13)  | 2966 (13) | 33 (14)   |
| 34 (14)        | 35 (14)   | 36 (14)   | 166 (14)   | 170 (14)  | 256 (14)   | 325 (14)  | 332 (14)  |
| 335 (14)       | 337 (14)  | 473 (14)  | 485 (14)   | 714 (14)  | 714 (14)   | 720 (14)  | 720 (14)  |
| 726 (14)       | 732 (14)  | 733 (14)  | 734 (14)   | 797 (14)  | 797 (14)   | 830 (14)  | 832 (14)  |
| 843 (14)       | 898 (14)  | 910 (14)  | 975 (14)   | 985 (14)  | 1020 (14)  | 1063 (14) | 1071 (14) |
| 1073 (14)      | 1075 (14) | 1207 (14) | 1232 (14)  | 1233 (14) | 1243 (14)  | 41 (15)   | 105 (15)  |
| 107 (15)       | 108 (15)  | 177 (15)  | 233 (15)   | 9 (16)    | 13 (16)    | 64 (16)   | 65 (16)   |
| 69 (16)        | 73 (16)   | 199 (16)  | 270 (16)   | 277 (16)  | 287 (16)   | 332 (16)  | 338 (16)  |
| 340 (16)       | 414 (16)  | 421 (16)  | 578 (16)   | 578 (16)  | 581 (16)   | 705 (16)  | 908 (16)  |
| 1119 (16)      | 1210 (16) | 1217 (16) | 1221 (16)  | 1237 (16) | 1358 (16)  | 1434 (16) | 1528 (16) |
| 1553 (16)      | 1956 (16) | 1957 (16) | 2027 (16)  | 2029 (16) | 2053 (16)  | 2140 (16) | 2195 (16) |
| 2252 (16)      | 2329 (16) | 2440 (16) | 2534 (16)  | 2535 (16) | 2645 (16)  | 2683 (16) | 2803 (16) |
| 2806 (16)      | 2807 (16) | 2926 (16) | 136 (17)   | 156 (17)  | 365 (17)   | 410 (17)  | 580 (17)  |
| 628 (17)       | 629 (17)  | 630 (17)  | 720 (17)   | 721 (17)  | 722 (17)   | 754 (17)  | 757 (17)  |
| 801 (17)       | 802 (17)  | 810 (17)  | 811 (17)   | 904 (17)  | 908 (17)   | 1041 (17) | 1063 (17) |
| 1219 (17)      | 1221 (17) | 1222 (17) | 1249 (17)  | 1289 (17) | 1316 (17)  | 1317 (17) | 1399 (17) |
| 1402 (17)      | 1464 (17) | 1465 (17) | 1486 (17)  | 112 (18)  | 467 (18)   | 579 (18)  | 695 (18)  |
| 746 (18)       | 766 (18)  | 1904 (18) | 19 (19)    | 20 (19)   | 88 (19)    | 89 (19)   | 399 (19)  |
| 555 (19)       | 556 (19)  | 941 (19)  | 941 (19)   | 946 (19)  | 947 (19)   | 1002 (19) | 1025 (19) |
| 296 (20)       | 551 (20)  | 554 (20)  | 12 (21)    | 164 (21)  | 378 (21)   | 504 (21)  | 534 (21)  |
| 624 (21)       | 721 (21)  | 731 (21)  | 750 (21)   | 798 (21)  | 816 (21)   | 852 (21)  | 870 (21)  |
| 924 (21)       | 1051 (21) | 1169 (21) | 1182 (21)  | 1344 (21) | 1404 (21)  | 1432 (21) | 121 (22)  |
| 136 (23)       | 40 (24)   | 100 (24)  | 141 (24)   | 150 (24)  | 211 (24)   | 132 (25)  | 159* (25) |
| 165* (25)      | 172 (25)  | 173 (25)  | 174 (25)   | 175 (25)  | 178 (25)   | 179 (25)  | 180 (25)  |
| 231 (25)       | 240 (25)  | 245 (25)  | 252 (25)   | 264 (25)  | 264 (25)   | 275 (25)  | 278 (25)  |
| 288 (25)       | 293 (25)  | 307 (25)  | 316 (25)   | 323 (25)  | 326 (25)   | 342 (25)  | 345 (25)  |
| 346 (25)       | 350 (25)  | 358 (25)  | 380 (25)   | 386 (25)  | 390 (25)   | 402 (25)  | 406 (25)  |
| 413 (25)       | 414 (25)  | 415 (25)  | 445 (25)   | 492 (25)  | 494 (25)   | 25 (26)   | 59 (26)   |
| 100 (26)       | 133 (26)  | 166 (26)  | 172 (26)   | 186 (26)  | 204 (26)   | 233 (26)  | 253 (26)  |
| 256 (26)       | 336 (26)  | 357 (26)  | 357 (26)   | 402 (26)  | 454 (26)   | 468 (26)  | 480 (26)  |
| 482 (26)       | 533 (26)  | 549 (26)  | 579 (26)   | 580 (26)  | 581 (26)   | 584 (26)  | 690 (26)  |
| 693 (26)       | 738 (26)  | 748 (26)  | 751 (26)   | 784 (26)  | 829 (26)   | 841 (26)  | 842 (26)  |
| 859 (26)       | 866* (26) | 873 (26)  | 905 (26)   | 1097 (26) | 1130 (26)  | 1341 (26) | 1353 (26) |
| 221 (27)       | 224 (27)  | 228 (27)  | 240 (27)   | 248 (27)  | 34 (28)    | 104 (28)  | 106 (28)  |
| 155 (28)       | 399 (28)  | 409 (28)  | 565 (28)   | 567 (28)  | 570 (28)   | 572 (28)  | 597 (28)  |
| 746 (28)       | 757 (28)  | 796 (28)  | 797 (28)   | 798 (28)  | 800 (28)   | 891 (28)  | 896 (28)  |
| 897 (28)       | 899 (28)  | 961 (28)  | 1013 (28)  | 150 (29)  | 181 (29)   | 193 (29)  | 221 (29)  |
| 231 (29)       | 253 (29)  | 258 (29)  | 12 (30)    | 13 (30)   | 14 (30)    | 26 (30)   | 32 (30)   |
| 34 (30)        | 55 (30)   | 61 (30)   | 100 (30)   | 102 (30)  | 104 (30)   | 105 (30)  | 106 (30)  |
| 150 (30)       | 151 (30)  | 176 (30)  | 182 (30)   | 183 (30)  | 184 (30)   | 190 (30)  | 192 (30)  |
| 194 (30)       | 196 (30)  | 245 (30)  | 270 (30)   | 295 (30)  | 310 (30)   | 316 (30)  | 317 (30)  |
| 355 (30)       | 358 (30)  | 405 (30)  | 406 (30)   | 434 (30)  | 470 (30)   | 519 (30)  | 541 (30)  |
| 561 (30)       | 577 (30)  | 579 (30)  | 598 (30)   | 50 (31)   | 98 (31)    | 115 (31)  | 130 (31)  |
| 133 (31)       | 139 (31)  | 153 (31)  | 12 (32)    | 10 (33)   | 61 (33)    | 65 (33)   | 41 (32)   |
| 105 (32)       | 106 (32)  | 152 (32)  | 156 (32)   | 192 (32)  | 241 (32)   | 255 (32)  | 291 (32)  |
| 64 (34)        | 77 (34)   | 78 (34)   | 85 (34)    | 86 (34)   | 95 (34)    | 96 (34)   | 12 (35)   |
| 19 (35)        | 21 (35)   | 29 (35)   | 31 (35)    | 63 (35)   | 63 (35)    | 98 (35)   | 143 (35)  |
| 177 (35)       | 133 (36)  | 138 (36)  | 240 (36)   | 249 (36)  | 252 (36)   | 252 (36)  | 279 (36)  |
| 301 (36)       | 301 (36)  | 319 (36)  | 325 (36)   | 327 (36)  | 342 (36)   | 345 (36)  | 348 (36)  |
| 427 (36)       | 468 (36)  | 505 (36)  | 60 (37)    | 61 (37)   | 354 (37)   | 355 (37)  | 440 (37)  |
| 499 (37)       | 588 (37)  | 589 (37)  | 716 (37)   | 769 (37)  | 777 (37)   | 876 (37)  | 971 (37)  |
| 974 (37)       | 975 (37)  | 1092 (37) | 1107 (37)  | 1118 (37) | 1120 (37)  | 1123 (37) | 1124 (37) |
| 1125 (37)      | 1224 (37) | 1227 (37) | 1250 (37)  | 1252 (37) | 1281 (37)  | 1371 (37) | 1373 (37) |
| 1375 (37)      | 1379 (37) | 1436 (37) | 1437 (37)  | 1440 (37) | 1442 (37)  | 1443 (37) | 1444 (37) |
| 1445 (37)      | 1446 (37) | 1569 (37) | 1686* (37) | 1690 (37) | 1844 (37)  | 1881 (37) | 2031 (37) |
| 2031 (37)      | 2151 (37) | 36 (38)   | 76 (38)    | 77 (38)   | 79 (38)    | 81 (38)   | 106 (38)  |
| 112 (38)       | 122 (38)  | 137 (38)  | 138 (38)   | 154 (38)  | 178 (38)   | 186 (38)  | 236 (38)  |
| 276 (38)       | 277 (38)  | 301 (38)  | 306 (38)   | 358 (38)  | 377 (38)   | 378 (38)  | 387 (38)  |
| 388 (38)       | 395 (38)  | 450 (38)  | 505 (38)   | 512 (38)  | 534 (38)   | 565 (38)  | 568 (38)  |
| 588 (38)       | 605 (38)  | 36 (39)   | 56 (39)    | 58 (39)   | 163 (39)   | 183 (39)  | 184 (39)  |
| 18 (40)        | 242 (40)  | 268 (40)  | 269 (40)   | 272 (40)  | 273 (40)   | 295 (40)  | 298 (40)  |
| 299 (40)       | 399 (40)  | 400 (40)  | 401 (40)   | 402 (40)  | 403 (40)   | 404 (40)  | 405 (40)  |
| 406 (40)       | 407 (40)  | 408 (40)  | 409 (40)   | 413 (40)  | 543 (40)   | 543 (40)  | 547 (40)  |
| 577 (40)       | 581 (40)  | 582 (40)  | 695 (40)   | 796 (40)  | 1078 (40)  | 1079 (40) | 1348 (40) |
| 1360 (40)      | 1362 (40) | 1363 (40) | 1577 (40)  | 1635 (40) | 15 (41)    | 163 (41)  | 217 (41)  |
| 437 (41)       | 520 (41)  | 612 (41)  | 701 (41)   | 705 (41)  | 706 (41)   | 707 (41)  | 788 (41)  |
| 790 (41)       | 816 (41)  | 817 (41)  | 911 (41)   | 939 (41)  |            |           |           |
| 494* (13)      | 2045 (13) |           |            |           |            |           |           |
| 493* (13)      | 1441 (13) | 1442 (13) | 256 (18)   | 257 (18)  |            |           |           |
| 34* (38)       |           |           |            |           |            |           |           |
| 34* (38)       |           |           |            |           |            |           |           |
| Integer256     |           |           |            |           |            |           |           |
| Integer8       |           |           |            |           |            |           |           |
| intEOLHook     |           |           |            |           |            |           |           |
| intHitTestHook |           |           |            |           |            |           |           |
| inThumb        |           |           |            |           |            |           |           |
| IntMultiply    |           |           |            |           |            |           |           |
| 1061 (21)      | 1062 (21) | 1357 (21) | 1359 (21)  |           |            |           |           |
| 458 (10)       | 459 (10)  | 465 (10)  | 890 (10)   | 901 (10)  | 987 (10)   | 998 (10)  | 1201 (10) |
| 1216 (10)      | 1226 (10) | 1285 (10) | 1300 (10)  | 1310 (10) | 1569 (10)  | 1583 (10) | 1600 (10) |
| 1669 (10)      | 1683 (10) | 1700 (10) | 1812 (10)  | 1813 (10) | 1819 (10)  | 103 (23)  | 278* (25) |

|                          |           |           |           |           |           |           |                     |
|--------------------------|-----------|-----------|-----------|-----------|-----------|-----------|---------------------|
| intWidthHook             | 543 (37)  | 544 (37)  | 555 (37)  | 556 (37)  | 557 (37)  | 558 (37)  |                     |
| INumberText              | 34*(38)   |           |           |           |           |           |                     |
| InUpButton               | 644*( 5)  | 3395*( 6) |           |           |           |           |                     |
| inUpButton               | 1056 (21) |           |           |           |           |           |                     |
| invalidate               | 562 (20)  | 582 (20)  | 21 (21)   |           |           |           |                     |
|                          | 604*( 5)  | 606*( 5)  | 3218*( 6) | 3221 ( 6) | 3231*( 6) | 3234 ( 6) | 600*( 9) 2693*(10)  |
|                          | 2695 (10) | 1566*(13) | 1568*(13) | 1574*(13) | 1578*(13) | 1809*(13) | 1813*(13) 1961*(13) |
|                          | 1967*(13) | 2120*(13) | 2190*(13) | 2573*(13) | 1429*(18) | 1436 (18) | 1442 (18) 1447 (18) |
|                          | 1551*(18) | 1564 (18) | 1576 (18) | 1708*(18) | 1729 (18) | 1738*(18) | 761*(19) 816*(19)   |
|                          | 839 (19)  | 374*(20)  | 377 (20)  | 397*(20)  | 415 (20)  | 424*(21)  | 429 (21) 430 (21)   |
|                          | 777*(21)  | 791 (21)  | 297*(23)  | 243*(38)  | 1258*(40) | 1265 (40) | 1278 (40)           |
| InvalidateCell           | 343*( 9)  | 1404*(10) |           |           |           |           |                     |
| InvalidateSelection      | 346*( 9)  | 1420*(10) |           |           |           |           |                     |
| InvalidRect              | 2476 ( 6) | 2624 ( 6) | 1608*(13) | 1342*(18) | 833 (19)  | 836 (19)  | 503 (20)            |
| InvalidVRect             | 925 (10)  | 1022 (10) | 1410 (10) | 1437 (10) | 1604 (10) | 1704 (10) | 1604*(13) 1082 (18) |
|                          | 1355*(18) |           |           |           |           |           |                     |
| InvalPageFeedback        | 431*(36)  | 680 (37)  | 1081*(37) | 1631 (37) | 1669 (37) |           |                     |
| InvalRect                | 1347 (18) | 1364 (18) |           |           |           |           |                     |
| InvalRgn                 | 1593 (18) | 510 (20)  |           |           |           |           |                     |
| InVertex                 | 1491 (10) | 1499 (10) |           |           |           |           |                     |
| inVertex                 | 76*( 9)   | 665 (10)  |           |           |           |           |                     |
| InvertRect               | 357 (21)  |           |           |           |           |           |                     |
| InvertRgn                | 583 (10)  | 589 (10)  | 602 (10)  |           |           |           |                     |
| inZoomIn                 | 1630 (16) | 2662 (16) |           |           |           |           |                     |
| inZoomOut                | 1630 (16) | 2664 (16) |           |           |           |           |                     |
| ioDirID                  | 530-(17)  | 656-(17)  | 197 (26)  |           |           |           |                     |
| ioFDirIndex              | 529-(17)  | 241-(26)  |           |           |           |           |                     |
| ioFlFndrInfo             | 250 (17)  | 546 (17)  | 547 (17)  | 556 (17)  | 666 (17)  | 667 (17)  |                     |
| ioFlMdDat                | 252 (17)  | 177 (26)  |           |           |           |           |                     |
| ioFlPyLen                | 1150 (17) | 1184 (17) |           |           |           |           |                     |
| ioFlRPyLen               | 1149 (17) | 1184 (17) |           |           |           |           |                     |
| ioFVersNum               | 528-(17)  | 696-(17)  | 141-(26)  | 240-(26)  |           |           |                     |
| ioMisc                   | 608-(26)  |           |           |           |           |           |                     |
| ioNamePtr                | 526-(17)  | 541-(17)  | 645 (17)  | 665 (17)  | 676 (17)  | 702 (17)  | 1113-(17) 1269-(17) |
|                          | 1349-(17) | 139-(26)  | 212-(26)  | 238-(26)  | 604-(26)  |           |                     |
| ioPermsn                 | 607-(26)  | 618-(26)  |           |           |           |           |                     |
| ioRefNum                 | 657 (26)  |           |           |           |           |           |                     |
| ioRefnum                 | 624 (26)  | 628 (26)  |           |           |           |           |                     |
| ioValBlkSiz              | 1124 (17) |           |           |           |           |           |                     |
| ioVersNum                | 606-(26)  |           |           |           |           |           |                     |
| ioVrBlk                  | 1121 (17) |           |           |           |           |           |                     |
| ioVolIndex               | 1115-(17) |           |           |           |           |           |                     |
| ioVRefNum                | 605-(26)  |           |           |           |           |           |                     |
| ioVRefnum                | 527-(17)  | 645 (17)  | 665 (17)  | 674 (17)  | 702 (17)  | 1114-(17) | 1270-(17) 1350-(17) |
|                          | 140-(26)  | 197 (26)  | 213-(26)  | 239-(26)  |           |           |                     |
| ioWDDirID                | 220 (26)  |           |           |           |           |           |                     |
| ioWDIndex                | 214-(26)  |           |           |           |           |           |                     |
| ioWDProcID               | 215-(26)  |           |           |           |           |           |                     |
| ioWDVRefnum              | 216-(26)  | 219 (26)  |           |           |           |           |                     |
| iPageH                   | 543 (37)  | 1759-(37) |           |           |           |           |                     |
| iPageV                   | 544 (37)  | 1758-(37) |           |           |           |           |                     |
| IPattern                 | 352*( 5)  | 1613*( 6) |           |           |           |           |                     |
| IPicture                 | 397*( 5)  | 1833*( 6) |           |           |           |           |                     |
| IPopup                   | 446*( 5)  | 2024*( 6) |           |           |           |           |                     |
| iPrAbort                 | 80 (37)   | 83 (37)   |           |           |           |           |                     |
| iPrAbort                 | 1194 (37) |           |           |           |           |           |                     |
| IPrintHandler            | 2339*(13) | 647 (14)  | 13*(24)   | 260 (37)  |           |           |                     |
| IPrintStyleChangeCommand | 491*(36)  | 1345 (37) | 2092*(37) |           |           |           |                     |
| IPrPgFract               | 543 (37)  | 544 (37)  |           |           |           |           |                     |
| iPrVersion               | 1745-(37) |           |           |           |           |           |                     |
| IRadio                   | 211*( 5)  | 989*( 6)  |           |           |           |           |                     |
| IRes                     | 73*( 5)   | 147*( 5)  | 177*( 5)  | 217*( 5)  | 260*( 5)  | 312*( 5)  | 358*( 5) 403*( 5)   |
|                          | 452*( 5)  | 519*( 5)  | 573*( 5)  | 649*( 5)  | 298*( 6)  | 316 ( 6)  | 796*( 6) 802 ( 6)   |
|                          | 878*( 6)  | 884 ( 6)  | 1006*( 6) | 1012 ( 6) | 1158*( 6) | 1162 ( 6) | 1433*( 6) 1448 ( 6) |
|                          | 1656*( 6) | 1671 ( 6) | 1870*( 6) | 1885 ( 6) | 2080*( 6) | 2096 ( 6) | 2708*( 6) 2712 ( 6) |
|                          | 2951*( 6) | 2955 ( 6) | 3415*( 6) | 3421 ( 6) | 193*( 9)  | 451*( 9)  | 140*(10) 146 (10)   |
|                          | 2234*(10) | 2239 (10) | 1482*(13) | 1776*(13) | 1936*(13) | 2078*(13) | 2159*(13) 2237*(13) |
|                          | 2291*(13) | 2565*(13) | 84 (15)   | 219*(18)  | 83*(19)   | 108 (19)  | 46*(20) 55 (20)     |
|                          | 127*(21)  | 135 (21)  | 552*(21)  | 556 (21)  | 962*(21)  | 968 (21)  | 1244*(21) 1255 (21) |
|                          | 42*(23)   | 48 (23)   | 160*(38)  | 70*(40)   | 87 (40)   |           |                     |
| IRowSelectCommand        | 714*( 9)  | 651 (10)  | 3079*(10) |           |           |           |                     |
| isActive                 | 1892*(13) | 986*(40)  | 994-(40)  | 995 (40)  | 1011 (40) |           |                     |
| isCellSelected           | 350*( 9)  | 1762*(10) | 1764-(10) | 2684 (10) |           |           |                     |
| isClosable               | 922*(13)  | 1894*(13) | 1211*(16) | 1269-(16) |           |           |                     |
| IScrollBar               | 2231*(13) | 941*(21)  | 1220 (21) |           |           |           |                     |
| IScroller                | 1769*(13) | 867 (14)  | 940 (14)  | 13*(20)   |           |           |                     |
| isDeskAccessory          | 758*(13)  | 544 (16)  | 1871*(16) | 1874-(16) | 1876-(16) | 2409 (16) | 3378 (16)           |
| isDialogEvent            | 75 (37)   |           |           |           |           |           |                     |
| isDimmed                 | 2112*(13) | 415*(21)  | 417-(21)  |           |           |           |                     |
| isDimmed                 | 2049*(13) | 144 (21)  | 182-(21)  |           |           |           |                     |
| isEnabled                | 1444*(13) | 156 (15)  | 227 (18)  | 259-(18)  |           |           |                     |
| isHilited                | 2050*(13) | 143 (21)  | 183-(21)  |           |           |           |                     |
| isItemSelected           | 597*( 9)  | 2677*(10) | 2684-(10) |           |           |           |                     |
| isModal                  | 1870*(13) | 1899*(13) | 132 (19)  | 190-(19)  |           |           |                     |
| isObject                 | 1108 (18) | 1111 (18) | 128 (24)  | 122*(31)  | 195 (32)  | 219*(32)  | 226-(32) 229-(32)   |
| isOn                     | 188*( 5)  | 228*( 5)  | 915 ( 6)  | 947*( 6)  | 949-( 6)  | 967 ( 6)  | 976 ( 6) 977 ( 6)   |
|                          | 1042 ( 6) | 1065 ( 6) | 1076*( 6) | 1078-( 6) | 1096 ( 6) | 1105 ( 6) | 1106 ( 6) 1333 ( 6) |
| isOn                     | 164*( 5)  | 204*( 5)  | 889 ( 6)  | 915-( 6)  | 1017 ( 6) | 1042-( 6) |                     |
| isOpen                   | 2053*(16) | 2067-(16) | 2072-(16) | 2102 (16) |           |           |                     |
| isResizable              | 922*(13)  | 1893*(13) | 1211*(16) | 1270-(16) |           |           |                     |
| isRevert                 | 803*(17)  | 823-(17)  | 824 (17)  | 839 (17)  | 840 (17)  |           |                     |
| ISScrollbar              | 2284*(13) | 229 (20)  | 241 (20)  | 1205*(21) |           |           |                     |
| isShown                  | 1659*(13) | 2002*(13) | 688 (18)  | 888 (18)  | 1372*(18) | 1374-(18) | 407 (19) 515 (19)   |
|                          | 524 (19)  | 747*(19)  | 751-(19)  | 753-(19)  |           |           |                     |
| isShown                  | 1443*(13) | 226 (18)  | 258-(18)  |           |           |           |                     |
| isSizable                | 2048*(13) | 142 (21)  | 181-(21)  |           |           |           |                     |
| IStaticText              | 513*( 5)  | 2668*( 6) | 2932 ( 6) |           |           |           |                     |
| IStdPrintHandler         | 192*(36)  | 201 (37)  | 245*(37)  |           |           |           |                     |
| isThisTheItem            | 736*(12)  | 738-(12)  | 745 (12)  |           |           |           |                     |



|                   |           |           |           |           |           |           |           |           |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| isTurnedOn        | 173*( 5)  | 213*( 5)  | 863*( 6)  | 867 ( 6)  | 991*( 6)  | 995 ( 6)  |           |           |
| isUndoable        | 445*(36)  | 1316*(37) | 1332 (37) |           |           |           |           |           |
| isViewEnabled     | 1655*(13) | 1287 (18) | 1331 (18) | 1381*(18) | 1383-(18) |           |           |           |
| italic            | 1035 (26) |           |           |           |           |           |           |           |
| itChanged         | 509*(37)  | 573 (37)  |           |           |           |           |           |           |
| ITECommand        | 400*(38)  | 783 (40)  | 13*(41)   | 323 (41)  | 456 (41)  | 527 (41)  | 645 (41)  |           |
| ITECutCopyCommand | 465*(38)  | 769 (40)  | 320*(41)  |           |           |           |           |           |
| item              | 480*( 5)  | 484*( 5)  | 672*( 6)  | 680 ( 6)  | 2428*( 6) | 2431 ( 6) | 2460*( 6) | 2464 ( 6) |
|                   | 2468 ( 6) | 2469 ( 6) | 2470 ( 6) | 91*(11)   | 115*(11)  | 132*(11)  | 154*(11)  | 184*(11)  |
|                   | 194*(11)  | 202*(11)  | 212*(11)  | 260*(11)  | 272*(11)  | 81*(12)   | 108*(12)  | 155*(12)  |
|                   | 169-(12)  | 170 (12)  | 183*(12)  | 196 (12)  | 208 (12)  | 237*(12)  | 242*(12)  | 281-(12)  |
|                   | 282 (12)  | 283 (12)  | 313*(12)  | 378*(12)  | 393 (12)  | 418*(12)  | 440 (12)  | 442 (12)  |
|                   | 457 (12)  | 472*(12)  | 476 (12)  | 483*(12)  | 487 (12)  | 510*(12)  | 677*(12)  | 690 (12)  |
|                   | 697 (12)  | 715*(12)  | 718 (12)  | 718 (12)  | 181*(29)  | 193*(29)  | 55*(30)   | 64 (30)   |
|                   | 65 (30)   | 68 (30)   | 74 (30)   | 81 (30)   | 83 (30)   | 89 (30)   | 100*(30)  | 111-(30)  |
|                   | 127-(30)  | 138-(30)  | 245*(30)  | 252 (30)  | 256 (30)  | 258 (30)  | 270*(30)  | 277 (30)  |
|                   | 282 (30)  | 284 (30)  | 285 (30)  | 434*(30)  | 451-(30)  | 455 (30)  | 457 (30)  | 460 (30)  |
|                   | 519*(30)  | 526 (30)  | 529 (30)  | 541*(30)  | 548 (30)  | 551 (30)  | 598*(30)  | 605 (30)  |
|                   | 608 (30)  | 60*(37)   | 77 (37)   | 79 (37)   |           |           |           |           |
| item1             | 51*( 1)   | 73*( 2)   | 76 ( 2)   | 78 ( 2)   | 277*(11)  | 664*(12)  |           |           |
| item2             | 51*( 1)   | 73*( 2)   | 76 ( 2)   | 78 ( 2)   | 277*(11)  | 664*(12)  |           |           |
| ItemNo            | 1398 (37) |           |           |           |           |           |           |           |
| itemNo            | 1373*(37) | 1385-(37) | 1390-(37) |           |           |           |           |           |
| itemNum           | 145*( 6)  | 195 ( 6)  | 211 ( 6)  | 233 ( 6)  | 233 ( 6)  | 236 ( 6)  | 236 ( 6)  |           |
| itemOffset        | 434*( 5)  | 2103 ( 6) | 2157-( 6) |           |           |           |           |           |
| itemsID           | 1221*(16) |           |           |           |           |           |           |           |
| itemText          | 1377*(37) | 1401 (37) | 1402 (37) | 1403 (37) | 1404 (37) |           |           |           |
| itemType          | 61*(37)   | 1375*(37) | 1398 (37) |           |           |           |           |           |
| ITEPasteCommand   | 492*(38)  | 776 (40)  | 431*(41)  |           |           |           |           |           |
| ITEStyleCommand   | 511*(38)  | 803 (40)  | 519*(41)  |           |           |           |           |           |
| ITETypingCommand  | 552*(38)  | 819 (40)  | 632*(41)  |           |           |           |           |           |
| ITEView           | 619 ( 6)  | 141*(38)  | 13*(40)   | 374 (41)  |           |           |           |           |
| ITextGridView     | 427*( 9)  | 2199*(10) | 2457 (10) |           |           |           |           |           |
| ITextListView     | 522*( 9)  | 2441*(10) |           |           |           |           |           |           |
| itsAcceptsChanges | 73*(38)   | 102 (40)  | 143-(40)  |           |           |           |           |           |
| itsAdornment      | 1596*(13) | 2045*(13) | 394*(18)  | 477 (18)  | 483 (18)  | 487 (18)  | 490 (18)  | 493 (18)  |
|                   | 506 (18)  | 508 (18)  | 510 (18)  | 512 (18)  | 141 (21)  | 179-(21)  |           |           |
| itsArea           | 799*( 6)  | 805 ( 6)  | 807 ( 6)  | 881*( 6)  | 887 ( 6)  | 889 ( 6)  | 1009*( 6) | 1015 ( 6) |
|                   | 1017 ( 6) |           |           |           |           |           |           |           |
| itsAutoWrap       | 72*(38)   | 156*(38)  | 19*(40)   | 42 (40)   | 104 (40)  | 142-(40)  |           |           |
| itsBounds         | 2170*(13) | 536*(21)  | 541 (21)  | 542 (21)  | 623*(21)  | 641 (21)  |           |           |
| itsCancelItemID   | 69*( 5)   | 256*( 6)  | 281 ( 6)  |           |           |           |           |           |
| itsChoice         | 108*( 5)  | 186*( 5)  | 226*( 5)  | 273*( 5)  | 447*( 6)  | 449 ( 6)  | 466 ( 6)  | 936*( 6)  |
|                   | 938 ( 6)  | 940 ( 6)  | 1063*( 6) | 1065 ( 6) | 1068 ( 6) | 1222*( 6) | 1235 ( 6) | 1238 ( 6) |
|                   | 1556*(13) | 746*(18)  | 749 (18)  |           |           |           |           |           |
| itsCmd            | 1231*(13) | 135*(17)  |           |           |           |           |           |           |
| itsCmdNumber      | 646*( 9)  | 2778*(10) | 2785 (10) | 816*(13)  | 833*(13)  | 838*(13)  | 842*(13)  | 852*(13)  |
|                   | 869*(13)  | 1198*(13) | 1248*(13) | 1284*(13) | 1292*(13) | 1301*(13) | 1316*(13) | 1332*(13) |
|                   | 1347*(13) | 2393*(13) | 2461*(13) | 329*(16)  | 349 (16)  | 408*(16)  | 427 (16)  | 684*(16)  |
|                   | 1883*(16) | 1885 (16) | 2130*(16) | 2150 (16) | 2926*(16) | 515*(17)  | 734*(17)  | 784*(17)  |
|                   | 790 (17)  | 901*(17)  | 918 (17)  | 940 (17)  | 1036*(17) | 1087 (17) | 1103 (17) | 1133 (17) |
|                   | 1173 (17) | 1206*(17) | 1247*(17) | 1257 (17) | 1287*(17) | 1345 (17) | 1399*(17) | 1408 (17) |
|                   | 12*(22)   | 15 (22)   | 160*(24)  | 339*(36)  | 1427*(37) | 1464 (37) | 185*(38)  | 400*(38)  |
|                   | 466*(38)  | 512*(38)  | 795*(40)  | 803 (40)  | 13*(41)   | 50 (41)   | 320*(41)  | 323 (41)  |
|                   | 325 (41)  | 520*(41)  | 527 (41)  |           |           |           |           |           |
| itsCreator        | 1140*(13) | 10*(17)   | 45 (17)   |           |           |           |           |           |
| itsCurrentItem    | 448*( 5)  | 2026*( 6) | 2046 ( 6) | 2054 ( 6) |           |           |           |           |
| itsDefItemID      | 69*( 5)   | 256*( 6)  | 280 ( 6)  |           |           |           |           |           |
| itsDialogView     | 3315*( 6) |           |           |           |           |           |           |           |
| itsDirection      | 2233*(13) | 2286*(13) | 943*(21)  | 947 (21)  | 949 (21)  | 1207*(21) | 1221 (21) |           |
| itsDocument       | 66*( 5)   | 73*( 5)   | 147*( 5)  | 177*( 5)  | 217*( 5)  | 260*( 5)  | 312*( 5)  | 358*( 5)  |
|                   | 403*( 5)  | 452*( 5)  | 519*( 5)  | 573*( 5)  | 649*( 5)  | 253*( 6)  | 273 ( 6)  | 298*( 6)  |
|                   | 316 ( 6)  | 796*( 6)  | 878*( 6)  | 1006*( 6) | 1158*( 6) | 1433*( 6) | 1656*( 6) | 1870*( 6) |
|                   | 2080*( 6) | 2708*( 6) | 2951*( 6) | 3415*( 6) | 172*( 9)  | 193*( 9)  | 428*( 9)  | 451*( 9)  |
|                   | 523*( 9)  | 30*(10)   | 101 (10)  | 140*(10)  | 146 (10)  | 2200*(10) | 2217 (10) | 2234*(10) |
|                   | 2239 (10) | 2442*(10) | 2457 (10) | 650*(13)  | 652*(13)  | 1475*(13) | 1482*(13) | 1776*(13) |
|                   | 1931*(13) | 1936*(13) | 2026*(13) | 2078*(13) | 2159*(13) | 2237*(13) | 2291*(13) | 2461*(13) |
|                   | 2565*(13) | 2852*(13) | 2861*(13) | 2865*(13) | 2871*(13) | 831*(14)  | 849 (14)  | 899*(14)  |
|                   | 916 (14)  | 975*(14)  | 1007 (14) | 1020*(14) | 1027 (14) | 57*(15)   | 65 (15)   | 84 (15)   |
|                   | 105*(15)  | 124 (15)  | 176 (15)  | 180 (15)  | 179*(18)  | 193 (18)  | 209 (18)  | 210 (18)  |
|                   | 219*(18)  | 224 (18)  | 15*(19)   | 71 (19)   | 83*(19)   | 147 (19)  | 719*(19)  | 725 (19)  |
|                   | 726 (19)  | 729 (19)  | 730 (19)  | 733 (19)  | 46*(20)   | 127*(21)  | 136 (21)  | 552*(21)  |
|                   | 556 (21)  | 962*(21)  | 968 (21)  | 1244*(21) | 1255 (21) | 12*(22)   | 16 (22)   | 42*(23)   |
|                   | 48 (23)   | 192*(36)  | 245*(37)  | 261 (37)  | 297-(37)  | 299 (37)  | 301 (37)  | 302 (37)  |
|                   | 303 (37)  | 304 (37)  | 305 (37)  | 319*(37)  | 326-(37)  | 327 (37)  | 330 (37)  | 331 (37)  |
|                   | 332 (37)  | 335 (37)  | 142*(38)  | 160*(38)  | 13*(40)   | 49 (40)   | 70*(40)   | 87 (40)   |
| itsDown           | 185*(39)  | 190-(39)  | 191 (39)  |           |           |           |           |           |
| itsExtent         | 1645*(13) | 1805*(13) | 1136*(18) | 1138 (18) | 1139 (18) | 328*(20)  | 331 (20)  | 332 (20)  |
| itsFileType       | 1140*(13) | 10*(17)   | 44 (17)   |           |           |           |           |           |
| itsFirstChar      | 553*(38)  | 632*(41)  | 656 (41)  |           |           |           |           |           |
| itsFontColor      | 113*( 9)  | 2245 (10) | 2290-(10) |           |           |           |           |           |
| itsFontFace       | 111*( 9)  | 2244 (10) | 2288-(10) |           |           |           |           |           |
| itsFontName       | 114*( 9)  | 2244 (10) | 2293 (10) | 2057*(13) | 148 (21)  | 193 (21)  | 83*(38)   | 99 (40)   |
|                   | 156 (40)  |           |           |           |           |           |           |           |
| itsFontSize       | 112*( 9)  | 2245 (10) | 2289-(10) |           |           |           |           |           |
| itsFrame          | 1649*(13) | 885*(18)  | 895 (18)  | 896 (18)  | 897 (18)  | 1146*(18) | 1148 (18) | 1149 (18) |
|                   | 1149 (18) | 1150 (18) | 1150 (18) |           |           |           |           |           |
| itsFreesText      | 74*(38)   | 105 (40)  | 144-(40)  |           |           |           |           |           |
| itsGrafPort       | 337*(18)  | 360-(18)  | 361 (18)  | 362 (18)  |           |           |           |           |
| itsHDeterminer    | 148*(38)  | 15*(40)   | 49 (40)   |           |           |           |           |           |
| itsHFixedSize     | 193*(36)  | 246*(37)  | 268 (37)  |           |           |           |           |           |
| itsHorzMax        | 1771*(13) | 15*(20)   | 32 (20)   |           |           |           |           |           |
| itsHSizeDet       | 68*( 5)   | 142*( 5)  | 172*( 5)  | 212*( 5)  | 255*( 5)  | 307*( 5)  | 353*( 5)  | 398*( 5)  |
|                   | 447*( 5)  | 514*( 5)  | 255*( 6)  | 273 ( 6)  | 781*( 6)  | 784 ( 6)  | 862*( 6)  | 865 ( 6)  |
|                   | 990*( 6)  | 993 ( 6)  | 1120*( 6) | 1136 ( 6) | 1391*( 6) | 1406 ( 6) | 1614*( 6) | 1629 ( 6) |
|                   | 1834*( 6) | 1850 ( 6) | 2025*( 6) | 2042 ( 6) | 2669*( 6) | 2685 ( 6) | 1442*(13) | 1479*(13) |
|                   | 1770*(13) | 2074*(13) | 2154*(13) | 2232*(13) | 2285*(13) | 154 (15)  | 180*(18)  | 196 (18)  |
|                   | 225 (18)  | 257-(18)  | 14*(20)   | 24 (20)   | 98*(21)   | 106 (21)  | 533*(21)  | 540 (21)  |
|                   | 942*(21)  | 945 (21)  | 1206*(21) | 1220 (21) |           |           |           |           |

|                  |           |           |           |           |           |           |           |           |  |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| itsID            | 2053*(16) | 2068 (16) | 2068 (16) |           |           |           |           |           |  |
| itsId            | 2070 (16) |           |           |           |           |           |           |           |  |
| itsIdentifier    | 118*( 5)  | 698*( 6)  | 704 ( 6)  | 1538*(13) | 947*(18)  | 958 (18)  | 966 (18)  | 976 (18)  |  |
| itsIndex         | 256*( 5)  | 515*( 5)  | 1121*( 5) | 1138 ( 6) | 2670*( 6) | 2687 ( 6) |           |           |  |
| itsInset         | 2053*(13) | 146 (21)  | 185-(21)  | 78*(38)   | 152*(38)  | 16*(40)   | 31 (40)   | 95 (40)   |  |
|                  | 147-(40)  |           |           |           |           |           |           |           |  |
| itsItemOffset    | 448*( 5)  | 2026*( 6) | 2047 ( 6) |           |           |           |           |           |  |
| itsJustification | 79*(38)   | 154*(38)  | 18*(40)   | 37 (40)   | 101 (40)  | 148-(40)  |           |           |  |
| itsKey           | 33*( 1)   | 13*( 2)   | 31 ( 2)   |           |           |           |           |           |  |
| itsKeyCmdNumber  | 76*(38)   | 96 (40)   | 145-(40)  |           |           |           |           |           |  |
| itsLabel         | 135*( 5)  | 143*( 5)  | 165*( 5)  | 173*( 5)  | 205*( 5)  | 213*( 5)  | 244*( 5)  | 782*( 6)  |  |
|                  | 785 ( 6)  | 807 ( 6)  | 834 ( 6)  | 863*( 6)  | 866 ( 6)  | 889 ( 6)  | 917 ( 6)  | 991*( 6)  |  |
|                  | 994 ( 6)  | 1017 ( 6) | 1044 ( 6) | 1166 ( 6) | 1193 ( 6) |           |           |           |  |
| itsLocation      | 67*( 5)   | 141*( 5)  | 171*( 5)  | 211*( 5)  | 254*( 5)  | 306*( 5)  | 352*( 5)  | 397*( 5)  |  |
|                  | 446*( 5)  | 513*( 5)  | 568*( 5)  | 644*( 5)  | 254*( 6)  | 273 ( 6)  | 780*( 6)  | 784 ( 6)  |  |
|                  | 861*( 6)  | 865 ( 6)  | 989*( 6)  | 993 ( 6)  | 1119*( 6) | 1136 ( 6) | 1390*( 6) | 1406 ( 6) |  |
|                  | 1613*( 6) | 1629 ( 6) | 1833*( 6) | 1850 ( 6) | 2024*( 6) | 2042 ( 6) | 2668*( 6) | 2685 ( 6) |  |
|                  | 2925*( 6) | 2932 ( 6) | 3395*( 6) | 3401 ( 6) | 174*( 9)  | 430*( 9)  | 525*( 9)  | 32*(10)   |  |
|                  | 101 (10)  | 2202*(10) | 2217 (10) | 2444*(10) | 2457 (10) | 1439*(13) | 1477*(13) | 1769*(13) |  |
|                  | 2073*(13) | 2153*(13) | 2231*(13) | 2284*(13) | 837*(14)  | 858-(14)  | 859 (14)  | 867 (14)  |  |
|                  | 152 (15)  | 179*(18)  | 194 (18)  | 224 (18)  | 254-(18)  | 13*(20)   | 24 (20)   | 97*(21)   |  |
|                  | 106 (21)  | 532*(21)  | 540 (21)  | 941*(21)  | 945 (21)  | 1205*(21) | 1220 (21) | 17*(23)   |  |
|                  | 21 (23)   | 26 (23)   | 146*(38)  | 14*(40)   | 49 (40)   |           |           |           |  |
| itsMainFileType  | 687*(13)  | 58*(16)   | 95 (16)   |           |           |           |           |           |  |
| itsMainView      | 2853*(13) | 831*(14)  | 869 (14)  | 872 (14)  | 874 (14)  |           |           |           |  |
| itsMargins       | 2928*( 6) | 352*(41)  | 369 (41)  | 377 (41)  |           |           |           |           |  |
| itsMax           | 2155*(13) | 2171*(13) | 2192*(13) | 2200*(13) | 2233*(13) | 2257*(13) | 2263*(13) | 2286*(13) |  |
|                  | 534*(21)  | 542 (21)  | 624*(21)  | 642 (21)  | 798*(21)  | 805 (21)  | 870*(21)  | 878 (21)  |  |
|                  | 943*(21)  | 946 (21)  | 948 (21)  | 1108*(21) | 1111 (21) | 1113 (21) | 1115 (21) | 1117-(21) |  |
|                  | 1117 (21) | 1120 (21) | 1157*(21) | 1160 (21) | 1207*(21) | 1221 (21) |           |           |  |
| itsMaxChars      | 569*( 5)  | 2926*( 6) | 2934 ( 6) | 77*(38)   | 97 (40)   | 146-(40)  |           |           |  |
| itsMaximum       | 645*( 5)  | 3396*( 6) | 3403 ( 6) |           |           |           |           |           |  |
| itsMin           | 2155*(13) | 2171*(13) | 2194*(13) | 2200*(13) | 2233*(13) | 2259*(13) | 2263*(13) | 534*(21)  |  |
|                  | 542 (21)  | 624*(21)  | 642 (21)  | 816*(21)  | 823 (21)  | 870*(21)  | 877 (21)  | 943*(21)  |  |
|                  | 946 (21)  | 948 (21)  | 1128*(21) | 1131 (21) | 1133 (21) | 1134 (21) | 1157*(21) | 1161 (21) |  |
| itsMinimum       | 645*( 5)  | 3396*( 6) | 3402 ( 6) |           |           |           |           |           |  |
| itsMode          | 186*(38)  | 512*(38)  | 796*(40)  | 803 (40)  | 520*(41)  | 532 (41)  |           |           |  |
| itsNewStyle      | 511*(38)  | 519*(41)  | 530 (41)  |           |           |           |           |           |  |
| itsNextHandler   | 593*(13)  | 12*(15)   | 14 (15)   |           |           |           |           |           |  |
| itsOpenCmd       | 859*(13)  | 2183*(16) | 2217 (16) | 2219 (16) |           |           |           |           |  |
| itsPaletteView   | 2854*(13) | 831*(14)  | 856 (14)  |           |           |           |           |           |  |
| itsAppFile       | 843*(13)  | 1883*(16) |           |           |           |           |           |           |  |
| itsParams        | 73*( 5)   | 77*( 5)   | 79*( 5)   | 147*( 5)  | 151*( 5)  | 153*( 5)  | 177*( 5)  | 181*( 5)  |  |
|                  | 183*( 5)  | 217*( 5)  | 221*( 5)  | 223*( 5)  | 260*( 5)  | 264*( 5)  | 266*( 5)  | 312*( 5)  |  |
|                  | 316*( 5)  | 318*( 5)  | 358*( 5)  | 362*( 5)  | 364*( 5)  | 403*( 5)  | 407*( 5)  | 409*( 5)  |  |
|                  | 452*( 5)  | 456*( 5)  | 458*( 5)  | 519*( 5)  | 523*( 5)  | 525*( 5)  | 573*( 5)  | 577*( 5)  |  |
|                  | 579*( 5)  | 649*( 5)  | 653*( 5)  | 655*( 5)  | 298*( 6)  | 316 ( 6)  | 317 ( 6)  | 335 ( 6)  |  |
|                  | 345*( 6)  | 351 ( 6)  | 353 ( 6)  | 367*( 6)  | 370 ( 6)  | 796*( 6)  | 802 ( 6)  | 806 ( 6)  |  |
|                  | 809 ( 6)  | 819*( 6)  | 826 ( 6)  | 830 ( 6)  | 844*( 6)  | 847 ( 6)  | 878*( 6)  | 884 ( 6)  |  |
|                  | 888 ( 6)  | 891 ( 6)  | 901*( 6)  | 908 ( 6)  | 912 ( 6)  | 924*( 6)  | 927 ( 6)  | 1006*( 6) |  |
|                  | 1012 ( 6) | 1016 ( 6) | 1018 ( 6) | 1028*( 6) | 1035 ( 6) | 1039 ( 6) | 1051*( 6) | 1054 ( 6) |  |
|                  | 1158*( 6) | 1162 ( 6) | 1165 ( 6) | 1168 ( 6) | 1178*( 6) | 1185 ( 6) | 1189 ( 6) | 1200*( 6) |  |
|                  | 1203 ( 6) | 1433*( 6) | 1448 ( 6) | 1450 ( 6) | 1471 ( 6) | 1481*( 6) | 1487 ( 6) | 1489 ( 6) |  |
|                  | 1509*( 6) | 1512 ( 6) | 1657*( 6) | 1671 ( 6) | 1673 ( 6) | 1693 ( 6) | 1703*( 6) | 1709 ( 6) |  |
|                  | 1711 ( 6) | 1728*( 6) | 1731 ( 6) | 1870*( 6) | 1885 ( 6) | 1887 ( 6) | 1897 ( 6) | 1907*( 6) |  |
|                  | 1913 ( 6) | 1915 ( 6) | 1928*( 6) | 1931 ( 6) | 2080*( 6) | 2096 ( 6) | 2100 ( 6) | 2129 ( 6) |  |
|                  | 2139*( 6) | 2145 ( 6) | 2147 ( 6) | 2165*( 6) | 2168 ( 6) | 2708*( 6) | 2712 ( 6) | 2715 ( 6) |  |
|                  | 2721 ( 6) | 2731*( 6) | 2738 ( 6) | 2742 ( 6) | 2757*( 6) | 2760 ( 6) | 2951*( 6) | 2955 ( 6) |  |
|                  | 2957 ( 6) | 2964 ( 6) | 2974*( 6) | 2980 ( 6) | 2982 ( 6) | 2995*( 6) | 2998 ( 6) | 3415*( 6) |  |
|                  | 3421 ( 6) | 3423 ( 6) | 3431 ( 6) | 3441*( 6) | 3447 ( 6) | 3449 ( 6) | 3464*( 6) | 3467 ( 6) |  |
|                  | 194*( 9)  | 199*( 9)  | 202*( 9)  | 452*( 9)  | 457*( 9)  | 460*( 9)  | 543*( 9)  | 140*(10)  |  |
|                  | 146 (10)  | 148 (10)  | 229 (10)  | 240*(10)  | 246 (10)  | 248 (10)  | 277*(10)  | 280 (10)  |  |
|                  | 2234*(10) | 2239 (10) | 2241 (10) | 2259 (10) | 2270*(10) | 2278 (10) | 2283 (10) | 2302*(10) |  |
|                  | 2305 (10) | 2472*(10) | 2475 (10) | 652*(13)  | 1482*(13) | 1484*(13) | 1486*(13) | 1776*(13) |  |
|                  | 1780*(13) | 1782*(13) | 1936*(13) | 1940*(13) | 1942*(13) | 2078*(13) | 2082*(13) | 2084*(13) |  |
|                  | 2159*(13) | 2161*(13) | 2163*(13) | 2237*(13) | 2241*(13) | 2243*(13) | 2291*(13) | 2295*(13) |  |
|                  | 2297*(13) | 2565*(13) | 57*(15)   | 65 (15)   | 68 (15)   | 84 (15)   | 90 (15)   | 219*(18)  |  |
|                  | 222 (18)  | 230 (18)  | 239*(18)  | 244 (18)  | 275*(18)  | 278 (18)  | 83*(19)   | 108 (19)  |  |
|                  | 113 (19)  | 160 (19)  | 171*(19)  | 178 (19)  | 182 (19)  | 214*(19)  | 217 (19)  | 46*(20)   |  |
|                  | 55 (20)   | 57 (20)   | 71 (20)   | 80*(20)   | 86 (20)   | 88 (20)   | 111*(20)  | 114 (20)  |  |
|                  | 127*(21)  | 135 (21)  | 138 (21)  | 151 (21)  | 162*(21)  | 169 (21)  | 174 (21)  | 204*(21)  |  |
|                  | 207 (21)  | 552*(21)  | 556 (21)  | 567*(21)  | 569 (21)  | 579*(21)  | 962*(21)  | 968 (21)  |  |
|                  | 972 (21)  | 981 (21)  | 991*(21)  | 997 (21)  | 999 (21)  | 1017*(21) | 1020 (21) | 1244*(21) |  |
|                  | 1255 (21) | 1273*(21) | 1275 (21) | 1285*(21) | 42*(23)   | 48 (23)   | 160*(38)  | 165*(38)  |  |
|                  | 167*(38)  | 70*(40)   | 87 (40)   | 89 (40)   | 117 (40)  | 127*(40)  | 132 (40)  | 164*(40)  |  |
|                  | 167 (40)  |           |           |           |           |           |           |           |  |
| itsparams        | 136 (40)  |           |           |           |           |           |           |           |  |
| itsParentID      | 1437*(13) | 149 (15)  | 162 (15)  | 164 (15)  | 166 (15)  | 172 (15)  | 250-(18)  | 252-(18)  |  |
| itsPenSize       | 1596*(13) | 2047*(13) | 394*(18)  | 451 (18)  | 451 (18)  | 481 (18)  | 481 (18)  | 485 (18)  |  |
|                  | 499 (18)  | 500 (18)  | 147 (21)  | 180-(21)  |           |           |           |           |  |
| itsPort          | 1550*(13) | 2168*(13) | 533*(18)  | 540 (18)  | 602*(21)  | 607 (21)  | 615 (21)  | 627*(21)  |  |
|                  | 637-(21)  | 638 (21)  | 639-(21)  | 641 (21)  | 646 (21)  | 649 (21)  | 227*(38)  | 214*(40)  |  |
|                  | 220 (40)  | 223 (40)  | 225 (40)  | 232 (40)  |           |           |           |           |  |
| itsPrintHandler  | 1671*(13) | 523*(18)  | 525 (18)  | 492*(36)  | 2092*(37) | 2105 (37) | 2108 (37) | 2113 (37) |  |
| itsProcID        | 2155*(13) | 2171*(13) | 534*(21)  | 542 (21)  | 624*(21)  | 642 (21)  |           |           |  |
| itsQDFrame       | 886*(18)  | 897 (18)  | 898 (18)  |           |           |           |           |           |  |
| itsRect          | 1387*(21) | 1393-(21) | 1394 (21) |           |           |           |           |           |  |
| itsRsrcID        | 256*( 5)  | 308*( 5)  | 354*( 5)  | 399*( 5)  | 448*( 5)  | 515*( 5)  | 1121*( 6) | 1137 ( 6) |  |
|                  | 1392*( 6) | 1408 ( 6) | 1615*( 6) | 1631 ( 6) | 1835*( 6) | 1851 ( 6) | 2026*( 6) | 2048 ( 6) |  |
|                  | 2051 ( 6) | 2054 ( 6) | 2670*( 6) | 2686 ( 6) | 650*(13)  | 2851*(13) | 2860*(13) | 2865*(13) |  |
|                  | 830*(14)  | 849 (14)  | 898*(14)  | 916 (14)  | 975*(14)  | 996 (14)  | 105*(15)  | 124 (15)  |  |
|                  | 129 (15)  |           |           |           |           |           |           |           |  |
| itsSaveText      | 400*(38)  | 13*(41)   | 53 (41)   |           |           |           |           |           |  |
| itsScroller      | 1548*(13) | 2287*(13) | 2462*(13) | 553*(18)  | 558 (18)  | 562 (18)  | 1208*(21) | 1229 (21) |  |
|                  | 1231 (21) | 1232 (21) | 13*(22)   | 26 (22)   | 230*(38)  | 240*(40)  | 244 (40)  | 245 (40)  |  |
|                  | 254 (40)  | 257 (40)  |           |           |           |           |           |           |  |
| itsSignature     | 1446*(13) | 70 (15)   | 147 (15)  | 174 (15)  | 260-(18)  |           |           |           |  |
| itsSize          | 67*( 5)   | 141*( 5)  | 171*( 5)  | 211*( 5)  | 254*( 5)  | 306*( 5)  | 352*( 5)  | 397*( 5)  |  |
|                  | 446*( 5)  | 513*( 5)  | 568*( 5)  | 644*( 5)  | 254*( 6)  | 273 ( 6)  | 780*( 6)  | 784 ( 6)  |  |

|                      |           |           |           |           |           |           |           |           |
|----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                      | 861*( 6)  | 865 ( 6)  | 989*( 6)  | 993 ( 6)  | 1119*( 6) | 1136 ( 6) | 1390*( 6) | 1406 ( 6) |
|                      | 1613*( 6) | 1629 ( 6) | 1833*( 6) | 1850 ( 6) | 2024*( 6) | 2042 ( 6) | 2668*( 6) | 2685 ( 6) |
|                      | 2925*( 6) | 2932 ( 6) | 3395*( 6) | 3401 ( 6) | 44*(10)   | 98 (10)   | 99 (10)   | 101 (10)  |
|                      | 1440*(13) | 1478*(13) | 1769*(13) | 2073*(13) | 2153*(13) | 2231*(13) | 2284*(13) | 836*(14)  |
|                      | 862 (14)  | 864 (14)  | 864 (14)  | 867 (14)  | 903*(14)  | 925=(14)  | 928 (14)  | 928 (14)  |
|                      | 934 (14)  | 934 (14)  | 940 (14)  | 153 (15)  | 179*(18)  | 195 (18)  | 224 (18)  | 255=(18)  |
|                      | 13*(20)   | 24 (20)   | 97*(21)   | 101 (21)  | 101 (21)  | 106 (21)  | 532*(21)  | 540 (21)  |
|                      | 941*(21)  | 945 (21)  | 1205*(21) | 1220 (21) | 18*(23)   | 22 (23)   | 26 (23)   | 153*(31)  |
|                      | 61*(33)   | 89 (33)   | 91 (33)   | 100 (33)  | 147*(38)  | 14*(40)   | 49 (40)   | 351*(41)  |
|                      | 368 (41)  | 375 (41)  |           |           |           |           |           |           |
| itsSquareDots        | 193*(36)  | 246*(37)  | 276 (37)  |           |           |           |           |           |
| itsStyleType         | 71*(38)   | 155*(38)  | 19*(40)   | 40 (40)   | 103 (40)  | 141=(40)  |           |           |
| itsSuperview         | 66*( 5)   | 73*( 5)   | 141*( 5)  | 147*( 5)  | 171*( 5)  | 177*( 5)  | 211*( 5)  | 217*( 5)  |
|                      | 254*( 5)  | 260*( 5)  | 306*( 5)  | 312*( 5)  | 352*( 5)  | 358*( 5)  | 397*( 5)  | 403*( 5)  |
|                      | 446*( 5)  | 452*( 5)  | 513*( 5)  | 519*( 5)  | 568*( 5)  | 573*( 5)  | 644*( 5)  | 649*( 5)  |
|                      | 253*( 6)  | 273 ( 6)  | 298*( 6)  | 316 ( 6)  | 780*( 6)  | 784 ( 6)  | 796*( 6)  | 802 ( 6)  |
|                      | 861*( 6)  | 865 ( 6)  | 878*( 6)  | 884 ( 6)  | 989*( 6)  | 993 ( 6)  | 1006*( 6) | 1012 ( 6) |
|                      | 1119*( 6) | 1136 ( 6) | 1158*( 6) | 1162 ( 6) | 1390*( 6) | 1406 ( 6) | 1433*( 6) | 1448 ( 6) |
|                      | 1613*( 6) | 1629 ( 6) | 1656*( 6) | 1671 ( 6) | 1833*( 6) | 1850 ( 6) | 1870*( 6) | 1885 ( 6) |
|                      | 2024*( 6) | 2042 ( 6) | 2080*( 6) | 2096 ( 6) | 2668*( 6) | 2685 ( 6) | 2708*( 6) | 2712 ( 6) |
|                      | 2925*( 6) | 2932 ( 6) | 2951*( 6) | 2955 ( 6) | 3395*( 6) | 3401 ( 6) | 3415*( 6) | 3421 ( 6) |
|                      | 173*( 9)  | 193*( 9)  | 429*( 9)  | 451*( 9)  | 524*( 9)  | 31*(10)   | 101 (10)  | 140*(10)  |
|                      | 146 (10)  | 219 (10)  | 220 (10)  | 2201*(10) | 2217 (10) | 2234*(10) | 2239 (10) | 2443*(10) |
|                      | 2457 (10) | 652*(13)  | 1482*(13) | 1769*(13) | 1776*(13) | 1936*(13) | 2073*(13) | 2078*(13) |
|                      | 2153*(13) | 2159*(13) | 2231*(13) | 2237*(13) | 2284*(13) | 2291*(13) | 2565*(13) | 57*(15)   |
|                      | 65 (15)   | 84 (15)   | 204 (18)  | 207 (18)  | 208 (18)  | 219*(18)  | 224 (18)  | 83*(19)   |
|                      | 108 (19)  | 13*(20)   | 24 (20)   | 46*(20)   | 55 (20)   | 97*(21)   | 106 (21)  | 127*(21)  |
|                      | 135 (21)  | 532*(21)  | 540 (21)  | 552*(21)  | 556 (21)  | 941*(21)  | 945 (21)  | 962*(21)  |
|                      | 968 (21)  | 1205*(21) | 1220 (21) | 1244*(21) | 1255 (21) | 42*(23)   | 48 (23)   | 145*(38)  |
|                      | 160*(38)  | 13*(40)   | 49 (40)   | 70*(40)   | 87 (40)   |           |           |           |
| itsSuperview         | 1476*(13) | 179*(18)  | 191 (18)  |           |           |           |           |           |
| itsTEView            | 400*(38)  | 465*(38)  | 492*(38)  | 511*(38)  | 552*(38)  | 13*(41)   | 28 (41)   | 29 (41)   |
|                      | 50 (41)   | 76 (41)   | 77 (41)   | 320*(41)  | 323 (41)  | 431*(41)  | 456 (41)  | 466 (41)  |
|                      | 489 (41)  | 519*(41)  | 527 (41)  | 529 (41)  | 632*(41)  | 645 (41)  |           |           |
| itsText              | 2196*(13) | 834*(21)  | 841 (21)  | 124*(41)  | 129=(41)  | 130 (41)  | 148 (41)  | 215*(41)  |
|                      | 223=(41)  | 224 (41)  | 237 (41)  |           |           |           |           |           |
| itsTextColor         | 2056*(13) | 148 (21)  | 190=(21)  | 82*(38)   | 99 (40)   | 153=(40)  |           |           |
| itsTextFace          | 2054*(13) | 148 (21)  | 188=(21)  | 80*(38)   | 99 (40)   | 151=(40)  |           |           |
| itsTextSize          | 2055*(13) | 148 (21)  | 189=(21)  | 81*(38)   | 99 (40)   | 152=(40)  |           |           |
| itsTextStyle         | 439*( 9)  | 532*( 9)  | 2211*(10) | 2213 (10) | 2236*(10) | 2244 (10) | 2247 (10) | 2374*(10) |
|                      | 2377=(10) | 2378 (10) | 2451*(10) | 2459 (10) | 153*(38)  | 17*(40)   | 36 (40)   |           |
| itsTitle             | 2155*(13) | 2170*(13) | 534*(21)  | 542 (21)  | 623*(21)  | 641 (21)  |           |           |
| itsType              | 1448*(13) | 69 (15)   | 73 (15)   | 90 (15)   | 262 (18)  |           |           |           |
| itsVal               | 2155*(13) | 2200*(13) | 2233*(13) | 2261*(13) | 2263*(13) | 534*(21)  | 542 (21)  | 870*(21)  |
|                      | 879 (21)  | 943*(21)  | 946 (21)  | 948 (21)  | 1142*(21) | 1145=(21) | 1145 (21) | 1146 (21) |
|                      | 1148 (21) | 1149 (21) | 1157*(21) | 1162 (21) |           |           |           |           |
| itsValue             | 33*( 1)   | 13*( 2)   | 34 ( 2)   | 645*( 5)  | 3396*( 6) | 3404 ( 6) | 2171*(13) | 624*(21)  |
|                      | 642 (21)  |           |           |           |           |           |           |           |
| itsVDeterminer       | 150*(38)  | 15*(40)   | 49 (40)   |           |           |           |           |           |
| itsVertMax           | 1771*(13) | 15*(20)   | 33 (20)   |           |           |           |           |           |
| itsVFixedSize        | 193*(36)  | 246*(37)  | 269 (37)  |           |           |           |           |           |
| itsView              | 647*( 9)  | 677*( 9)  | 714*( 9)  | 737*( 9)  | 760*( 9)  | 2779*(10) | 2785 (10) | 2785 (10) |
|                      | 2791 (10) | 2853*(10) | 2856 (10) | 3079*(10) | 3082 (10) | 3112*(10) | 3115 (10) | 3146*(10) |
|                      | 3149 (10) | 2339*(13) | 2462*(13) | 2861*(13) | 899*(14)  | 943 (14)  | 944 (14)  | 947 (14)  |
|                      | 948 (14)  | 950 (14)  | 951 (14)  | 13*(22)   | 25 (22)   | 13*(24)   | 15 (24)   | 192*(36)  |
|                      | 245*(37)  | 260 (37)  | 287 (37)  |           |           |           |           |           |
| itsVSizeDet          | 68*( 5)   | 142*( 5)  | 172*( 5)  | 212*( 5)  | 255*( 5)  | 307*( 5)  | 353*( 5)  | 398*( 5)  |
|                      | 447*( 5)  | 514*( 5)  | 255*( 6)  | 273 ( 6)  | 781*( 6)  | 784 ( 6)  | 862*( 6)  | 865 ( 6)  |
|                      | 990*( 6)  | 993 ( 6)  | 1120*( 6) | 1136 ( 6) | 1391*( 6) | 1406 ( 6) | 1614*( 6) | 1629 ( 6) |
|                      | 1834*( 6) | 1850 ( 6) | 2025*( 6) | 2042 ( 6) | 2669*( 6) | 2685 ( 6) | 1441*(13) | 1480*(13) |
|                      | 1770*(13) | 2074*(13) | 2154*(13) | 2232*(13) | 2285*(13) | 155 (15)  | 180*(18)  | 197 (18)  |
|                      | 225 (18)  | 256=(18)  | 14*(20)   | 24 (20)   | 98*(21)   | 106 (21)  | 533*(21)  | 540 (21)  |
|                      | 942*(21)  | 945 (21)  | 1206*(21) | 1220 (21) |           |           |           |           |
| itsWindow            | 512*( 6)  | 525=( 6)  | 526 ( 6)  | 527 ( 6)  | 646*( 6)  | 649=( 6)  | 650 ( 6)  | 652 ( 6)  |
|                      | 3335*( 6) | 3340=( 6) | 3341 ( 6) | 3342 ( 6) | 1105*(18) | 1110=(18) | 1111 (18) | 1111 (18) |
|                      | 1113 (18) | 1749*(18) | 1752=(18) | 1753 (18) | 1755 (18) | 1757 (18) |           |           |
| itsWMgrWindow        | 39 (19)   |           |           |           |           |           |           |           |
| itsWMgrWindow        | 1931*(13) | 15*(19)   | 34 (19)   | 43 (19)   | 44 (19)   | 65 (19)   | 67 (19)   |           |
| IVertexSelectCommand | 760*( 9)  | 669 (10)  | 3146*(10) |           |           |           |           |           |
| IView                | 273 ( 6)  | 101 (10)  | 1475*(13) | 179*(18)  | 224 (18)  | 41 (19)   | 24 (20)   | 106 (21)  |
|                      | 26 (23)   | 49 (40)   |           |           |           |           |           |           |
| ivRes                | 550 (37)  | 1750=(37) |           |           |           |           |           |           |
| IWindow              | 1931*(13) | 1007 (14) | 15*(19)   |           |           |           |           |           |
| -J-                  |           |           |           |           |           |           |           |           |
| j                    | 771*(10)  | 787=(10)  | 792 (10)  | 805 (10)  | 1084*(10) | 1108=(10) | 1112 (10) | 402*(26)  |
|                      | 406=(26)  | 408 (26)  |           |           |           |           |           |           |
| jmp                  | 12*(35)   | 21*(35)   | 31*(35)   | 116=(35)  | 160=(35)  | 195=(35)  |           |           |
| jmpPtr               | 63*(34)   | 83=(35)   | 237 (35)  | 239 (35)  | 240=(35)  |           |           |           |
| jmpRec               | 308 (30)  | 309*(30)  |           |           |           |           |           |           |
| jt                   | 571*(28)  |           |           |           |           |           |           |           |
| jtCount              | 897*(28)  |           |           |           |           |           |           |           |
| jtOffset             | 896*(28)  |           |           |           |           |           |           |           |
| jumpTablePtr         | 24*(12)   | 147*(31)  | 153*(31)  | 11*(33)   | 12*(33)   | 61*(33)   | 77 (33)   | 108 (33)  |
|                      | 135*(33)  | 139 (33)  |           |           |           |           |           |           |
| just                 | 501*( 5)  | 537*( 5)  | 600*( 5)  | 2639 ( 6) | 2717 ( 6) | 2747=( 6) | 2846*( 6) | 2848 ( 6) |
|                      | 3175*( 6) | 3186 ( 6) | 3189=( 6) | 3194=( 6) | 3196 ( 6) | 3197=( 6) | 3201=( 6) | 3201 ( 6) |
|                      | 3202 ( 6) | 425 (40)  |           |           |           |           |           |           |
| justification        | 405*(40)  | 425=(40)  | 466 (40)  | 514 (40)  |           |           |           |           |
| justLocked           | 341*(27)  | 1059*(28) | 1078 (28) |           |           |           |           |           |
| justSpool            | 326*(36)  | 1119*(37) | 1190 (37) | 1450*(37) | 1464=(37) | 1483 (37) | 1508 (37) |           |
| -K-                  |           |           |           |           |           |           |           |           |
| kAdorn               | 52*( 9)   |           |           |           |           |           |           |           |
| kEqualB              | 152 ( 2)  | 32*(11)   |           |           |           |           |           |           |
| kGreaterThanB        | 79 ( 2)   | 150 ( 2)  | 33*(11)   |           |           |           |           |           |
| kLessThanB           | 77 ( 2)   | 148 ( 2)  | 31*(11)   |           |           |           |           |           |
| kApplFontName        | 75*(25)   | 262 (26)  | 376 (26)  |           |           |           |           |           |
| kAskForFilename      | 454*(13)  |           |           |           |           |           |           |           |

[illegible]

|                        |           |           |           |           |           |           |           |           |  |
|------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| kPrintInfoSize         | 422*(13)  | 355 (17)  | 374 (17)  | 378 (17)  | 423 (17)  |           |           |           |  |
| kReadingFile           | 475*(13)  |           |           |           |           |           |           |           |  |
| kRedraw                | 59 (6)    | 61 (6)    | 531 (6)   | 939 (6)   | 1067 (6)  | 1231 (6)  | 2199 (6)  | 480*(13)  |  |
|                        | 384 (19)  | 806 (19)  | 543 (20)  | 584 (20)  | 654 (20)  | 475 (21)  | 477 (21)  | 481 (21)  |  |
|                        | 1334 (21) | 36 (22)   | 358 (40)  | 1331 (40) | 255 (41)  | 287 (41)  | 544 (41)  | 558 (41)  |  |
|                        | 860 (41)  |           |           |           |           |           |           |           |  |
| kRsrcCheckInterval     | 23*(14)   | 541 (14)  | 1133 (16) |           |           |           |           |           |  |
| kRsrcFileOverhead      | 431*(13)  | 357 (17)  |           |           |           |           |           |           |  |
| kRsrcOpen              | 471*(13)  |           |           |           |           |           |           |           |  |
| kRsrcOverhead          | 434*(13)  |           |           |           |           |           |           |           |  |
| kRsrcTypeOverhead      | 433*(13)  |           |           |           |           |           |           |           |  |
| kSaveCurrentChars      | 60*(38)   |           |           |           |           |           |           |           |  |
| kSBarSize              | 413*(13)  | 414 (13)  | 170 (20)  | 226 (20)  | 238 (20)  |           |           |           |  |
| kSBarSizeMinus1        | 414*(13)  | 862 (14)  | 863 (14)  | 928 (14)  | 930 (14)  | 934 (14)  | 936 (14)  | 479 (19)  |  |
|                        | 480 (19)  | 829 (19)  | 829 (19)  |           |           |           |           |           |  |
| kSelect                | 62*(9)    | 1955 (10) | 3040 (10) | 3042 (10) |           |           |           |           |  |
| kShowCantUndo          | 465*(13)  | 2866 (16) |           |           |           |           |           |           |  |
| kShowRedo              | 464*(13)  | 2880 (16) |           |           |           |           |           |           |  |
| kShowUndo              | 463*(13)  | 465 (13)  | 668 (14)  | 2870 (16) | 2878 (16) |           |           |           |  |
| kSlopToAllow           | 395*(40)  | 527 (40)  |           |           |           |           |           |           |  |
| kSpaces                | 161 (8)   | 80*(25)   | 341 (26)  | 405 (26)  | 414 (26)  | 415 (26)  | 443 (26)  | 181 (33)  |  |
|                        | 109 (32)  |           |           |           |           |           |           |           |  |
| kStdScrollUnit         | 411*(13)  | 30 (20)   | 30 (20)   | 252 (40)  | 254 (40)  |           |           |           |  |
| kStdSFWhere            | 2930*(16) | 2934 (16) | 1401*(17) | 1406 (17) |           |           |           |           |  |
| kStdStaggerAmount      | 420*(13)  | 156 (19)  | 156 (19)  |           |           |           |           |           |  |
| kStdSzMinus1SBar       | 418*(13)  |           |           |           |           |           |           |           |  |
| kStdSzSBar             | 416*(13)  | 418 (13)  |           |           |           |           |           |           |  |
| kSuspendOrResume       | 484*(13)  | 1670 (16) | 2621 (16) |           |           |           |           |           |  |
| kSwitchToTarget        | 456*(13)  | 205 (17)  |           |           |           |           |           |           |  |
| kSysClear              | 17*(14)   | 563 (14)  |           |           |           |           |           |           |  |
| kSysCopy               | 15*(14)   | 559 (14)  |           |           |           |           |           |           |  |
| kSysCut                | 14*(14)   | 557 (14)  |           |           |           |           |           |           |  |
| kSysFontName           | 74*(25)   | 260 (26)  | 368 (26)  |           |           |           |           |           |  |
| kSysPaste              | 16*(14)   | 561 (14)  |           |           |           |           |           |           |  |
| kSysUndo               | 13*(14)   | 555 (14)  |           |           |           |           |           |           |  |
| kTextGridView          | 38*(9)    |           |           |           |           |           |           |           |  |
| kTextListView          | 39*(9)    |           |           |           |           |           |           |           |  |
| kTopPalette            | 460*(13)  |           |           |           |           |           |           |           |  |
| kUnlimited             | 55*(38)   | 33 (40)   |           |           |           |           |           |           |  |
| kUseImmediateSuperView | 2781*(10) | 2785 (10) |           |           |           |           |           |           |  |
| kUsesDataFork          | 468*(13)  |           |           |           |           |           |           |           |  |
| kUsesRsrcFork          | 469*(13)  |           |           |           |           |           |           |           |  |
| kUsualPages            | 70*(36)   | 1643 (37) |           |           |           |           |           |           |  |
| kViewRsrcExpandAmt     | 436*(13)  | 48 (18)   | 122 (18)  |           |           |           |           |           |  |
| kVisible               | 3349 (6)  | 477*(13)  |           |           |           |           |           |           |  |
| kWantHScrollBar        | 450*(13)  |           |           |           |           |           |           |           |  |
| kWantVScrollBar        | 451*(13)  |           |           |           |           |           |           |           |  |
| kWatchDelay            | 68*(3)    | 122 (4)   | 138 (4)   | 158 (4)   |           |           |           |           |  |
| kWholeRect             | 66*(9)    | 2905 (10) |           |           |           |           |           |           |  |
| kWithoutStyle          | 620 (6)   | 59*(38)   | 1427 (40) | 585 (41)  |           |           |           |           |  |
| kWithStyle             | 58*(38)   | 310 (40)  | 596 (40)  | 657 (40)  | 1127 (40) | 1224 (40) | 1371 (40) | 1426 (40) |  |
|                        | 1493 (40) | 1541 (40) | 1605 (40) | 76 (41)   | 140 (41)  | 229 (41)  | 390 (41)  | 466 (41)  |  |
|                        | 489 (41)  | 735 (41)  | 916 (41)  |           |           |           |           |           |  |
| kWordAlign             | 77*(25)   |           |           |           |           |           |           |           |  |
| kYesButton             | 406*(13)  | 203 (17)  | 333 (17)  | 1158 (17) |           |           |           |           |  |
| -L-                    |           |           |           |           |           |           |           |           |  |
| labelRect              | 2246*(6)  | 2255 (6)  | 2265 (6)  | 2269 (6)  | 2269 (6)  | 2275 (6)  | 2341*(6)  | 2345 (6)  |  |
|                        | 2346 (6)  | 2346 (6)  | 2347 (6)  | 2350 (6)  |           |           |           |           |  |
| labelSlop              | 2338*(6)  | 2346 (6)  |           |           |           |           |           |           |  |
| labelWd                | 1248*(6)  | 1284*(6)  | 1285 (6)  |           |           |           |           |           |  |
| last                   | 104*(11)  | 495*(12)  | 500-(12)  | 502-(12)  | 1478 (18) |           |           |           |  |
| last                   | 120*(5)   | 592*(6)   | 597 (6)   | 713*(6)   | 726-(6)   | 740-(6)   | 745 (6)   | 107*(39)  |  |
|                        | 124-(39)  | 125 (39)  | 126 (39)  | 141-(39)  | 142 (39)  | 143 (39)  |           |           |  |
| lastAttempted          | 1444*(37) | 1509 (37) | 1511 (37) | 1514 (37) |           |           |           |           |  |
| lastChar               | 276*(38)  | 543*(40)  | 554 (40)  |           |           |           |           |           |  |
| lastCol                | 1088*(10) | 1095-(10) | 1101 (10) | 1110 (10) |           |           |           |           |  |
| lastDownPt             | 622*(13)  | 262*(15)  | 265 (15)  | 267 (15)  | 268 (15)  |           |           |           |  |
| lastHierMenu           | 196*(30)  | 224-(30)  | 225 (30)  |           |           |           |           |           |  |
| lastIsCR               | 297*(40)  | 306-(40)  | 316 (40)  | 319-(40)  | 331 (40)  |           |           |           |  |
| lastIsReturn           | 412*(40)  | 429-(40)  | 440 (40)  | 465 (40)  |           |           |           |           |  |
| lastPage               | 1437*(37) | 1472-(37) | 1474 (37) | 1478 (37) | 1518 (37) |           |           |           |  |
| lastPageTried          | 327*(36)  | 1120*(37) | 1144-(37) | 1170-(37) |           |           |           |           |  |
| lastParent             | 113*(15)  | 139-(15)  | 163-(15)  | 166-(15)  | 168 (15)  | 176 (15)  | 180 (15)  |           |  |
| lastParentID           | 112*(15)  | 138-(15)  | 164 (15)  | 172-(15)  |           |           |           |           |  |
| lastPrinted            | 1443*(37) | 1498-(37) | 1505 (37) | 1511-(37) | 1518 (37) |           |           |           |  |
| lastPtr                | 2931*(13) | 64*(18)   | 68 (18)   |           |           |           |           |           |  |
| lastRow                | 1086*(10) | 1094-(10) | 1102 (10) | 1108 (10) |           |           |           |           |  |
| LastSelectedCell       | 353*(9)   | 1773*(10) | 1775-(10) |           |           |           |           |           |  |
| LastSubviewThat        | 1525*(13) | 1284 (18) | 1327 (18) | 1390*(18) | 1393-(18) | 1395-(18) |           |           |  |
| LastThat               | 154*(11)  | 510*(12)  | 526-(12)  | 527 (12)  | 531-(12)  | 1393 (18) |           |           |  |
| LaunchClipboard        | 1032*(13) | 1892*(16) | 2693 (16) |           |           |           |           |           |  |
| lead                   | 183*(39)  | 203-(39)  | 208 (39)  |           |           |           |           |           |  |
| leading                | 1265 (6)  | 2360 (10) | 2361 (10) | 326 (40)  | 451 (40)  | 1417 (40) |           |           |  |
| left                   | 2197 (6)  | 2214 (6)  | 2230-(6)  | 2230 (6)  | 2381 (6)  | 2402 (6)  | 2408 (6)  | 2622 (6)  |  |
|                        | 2622 (6)  | 2625-(6)  | 2625 (6)  | 418-(10)  | 418 (10)  | 474-(10)  | 516 (10)  | 520 (10)  |  |
|                        | 731 (10)  | 733 (10)  | 736-(10)  | 773*(10)  | 777-(10)  | 777 (10)  | 779-(10)  | 779 (10)  |  |
|                        | 794-(10)  | 794 (10)  | 799 (10)  | 801 (10)  | 809-(10)  | 869-(10)  | 966-(10)  | 1474 (10) |  |
|                        | 1828-(10) | 1869 (10) | 1870 (10) | 2323 (10) | 2891-(10) | 2891 (10) | 2893 (10) | 2898-(10) |  |
|                        | 1032 (14) | 3044 (16) | 3065 (16) | 3065 (16) | 499 (18)  | 507 (18)  | 509 (18)  | 511 (18)  |  |
|                        | 1045 (18) | 1149 (18) | 363-(19)  | 366 (19)  | 479-(19)  | 567 (19)  | 568 (19)  | 588 (19)  |  |
|                        | 602 (19)  | 603 (19)  | 604 (19)  | 605 (19)  | 616 (19)  | 617 (19)  | 805 (19)  | 882 (19)  |  |
|                        | 968 (19)  | 976 (19)  | 1014 (19) | 225 (20)  | 226 (20)  | 264 (21)  | 783 (21)  | 784 (21)  |  |
|                        | 785 (21)  | 976 (21)  | 102 (23)  | 166*(25)  | 677 (26)  | 677 (26)  | 730 (26)  | 730 (26)  |  |
|                        | 988 (26)  | 1053 (26) | 369 (37)  | 543 (37)  | 555 (37)  | 919 (37)  | 1781 (37) | 283 (40)  |  |
|                        | 381 (40)  | 516-(40)  | 516 (40)  | 518-(40)  | 521-(40)  | 526-(40)  | 526 (40)  | 527 (40)  |  |
|                        | 528-(40)  | 533 (40)  | 1066-(40) | 1066 (40) | 1066 (40) | 1320 (40) | 1561 (40) | 50*(42)   |  |
| leftSlop               | 2362*(6)  | 2381 (6)  | 2402 (6)  |           |           |           |           |           |  |
| len                    | 2943*(13) | 137*(18)  | 139 (18)  |           |           |           |           |           |  |

|                      |            |            |            |            |            |            |            |            |
|----------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| LENGTH               | 682 ( 6)   | 682 ( 6)   | 831 ( 6)   | 913 ( 6)   | 1040 ( 6)  | 1190 ( 6)  | 1288 ( 6)  | 2350 ( 6)  |
|                      | 2385 ( 6)  | 2391 ( 6)  | 2743 ( 6)  | 2820 ( 6)  | 3520 ( 6)  | 3525 ( 6)  | 2284 (10)  | 1181 (14)  |
|                      | 69 (15)    | 90 (18)    | 245 (18)   | 183 (19)   | 175 (21)   | 52 (26)    | 1133 (26)  | 817 (28)   |
|                      | 163 (32)   | 163 (32)   | 162 (37)   | 531 (37)   | 137 (40)   | 1443 (40)  |            |            |
| Length               | 593 (12)   | 1105 (14)  | 1251 (14)  | 2166 (16)  | 591 (17)   |            |            |            |
| length               | 536* ( 5)  | 599* ( 5)  | 2846* ( 6) | 2848 ( 6)  | 3175* ( 6) | 3184 ( 6)  | 3208 ( 6)  | 155* (23)  |
|                      | 211* (23)  | 214 (23)   | 215 (23)   |            |            |            |            |            |
| LengthRect           | 363 (19)   | 365 (19)   | 288* (25)  | 468* (26)  | 471* (26)  | 1321 (40)  |            |            |
| LengthVRect          | 443 (20)   | 64* (42)   |            |            |            |            |            |            |
| lenTab               | 732* (14)  | 742* (14)  | 747 (14)   |            |            |            |            |            |
| lh                   | 467* (18)  | 469 (18)   | 471 (18)   |            |            |            |            |            |
| lhElements           | 1490* (40) | 1498* (40) | 1504 (40)  |            |            |            |            |            |
| LHHandle             | 579 (40)   | 1490 (40)  |            |            |            |            |            |            |
| lhHeight             | 613 (40)   |            |            |            |            |            |            |            |
| lhTab                | 579* (40)  | 610* (40)  | 610 (40)   | 613 (40)   | 1498 (40)  | 1504* (40) |            |            |
| lLimit               | 675* ( 6)  | 679* ( 6)  | 683 ( 6)   |            |            |            |            |            |
| Line                 | 471 (18)   | 472 (18)   |            |            |            |            |            |            |
| lineHeight           | 2592 ( 6)  | 103 (23)   | 65 (39)    | 247 (40)   | 248 (40)   | 332 (40)   | 408* (40)  | 436 (40)   |
|                      | 437 (40)   | 469 (40)   | 534 (40)   | 581* (40)  | 613* (40)  | 614 (40)   | 615 (40)   | 659 (40)   |
|                      | 659 (40)   | 1417* (40) |            |            |            |            |            |            |
| lineNo               | 582* (40)  | 601* (40)  | 606* (40)  | 611 (40)   | 613 (40)   | 618* (40)  | 618 (40)   | 620 (40)   |
|                      | 625 (40)   |            |            |            |            |            |            |            |
| lineStarts           | 461 (40)   | 466 (40)   | 470 (40)   | 472 (40)   |            |            |            |            |
| LineTo               | 2409 ( 6)  | 2410 ( 6)  | 792 (37)   |            |            |            |            |            |
| lineWidth            | 36* (14)   | 442 (14)   |            |            |            |            |            |            |
| LintToHex            | 130 (24)   | 292* (25)  | 479* (26)  | 539 (26)   | 559 (26)   | 1013 (26)  | 1345 (26)  | 1368 (26)  |
|                      | 307 (32)   |            |            |            |            |            |            |            |
| LintToHex            | 1200 (26)  | 205 (32)   |            |            |            |            |            |            |
| list                 | 39* (12)   | 41 (12)    | 42 (12)    | 44 (12)    | 45 (12)    | 956* (28)  | 964 (28)   | 966 (28)   |
| LoadMacAppSegment    | 228* (27)  | 746* (28)  | 765* (28)  |            |            |            |            |            |
| LoadResidentSegments | 349 (14)   | 236* (27)  | 793* (28)  |            |            |            |            |            |
| LoadResource         | 1542 ( 6)  | 1762 ( 6)  | 1960 ( 6)  |            |            |            |            |            |
| LoadScrap            | 533 (16)   |            |            |            |            |            |            |            |
| loc                  | 1699* (13) | 2376* (13) | 2405* (13) | 767* (18)  | 769 (18)   | 154* (20)  | 164* (20)  | 166 (20)   |
|                      | 166 (20)   | 167 (20)   | 167 (20)   | 173 (20)   | 173 (20)   | 101* (24)  | 141* (24)  | 232* (36)  |
|                      | 240* (36)  | 320* (36)  | 428* (36)  | 445* (37)  | 725* (37)  | 728 (37)   | 734 (37)   | 735 (37)   |
|                      | 770* (37)  | 783 (37)   | 840* (37)  | 844* (37)  | 854* (37)  | 857 (37)   | 859 (37)   | 860 (37)   |
|                      | 861* (37)  | 861 (37)   | 865 (37)   | 971* (37)  | 980* (37)  | 983* (37)  | 989* (37)  | 994* (37)  |
|                      | 997* (37)  | 997 (37)   | 1001 (37)  | 1092* (37) | 1097* (37) | 1098 (37)  |            |            |
| localPoint           | 1632* (13) | 841* (18)  | 214* (38)  | 911* (40)  |            |            |            |            |
| LocalToGlobal        | 169 ( 6)   | 170 ( 6)   | 2270 ( 6)  | 3016 (16)  | 637 (19)   | 638 (19)   |            |            |
| LocalToSuper         | 1502* (13) | 1799* (13) | 1402* (18) | 364* (20)  |            |            |            |            |
| LocalToWindow        | 1510* (13) | 1411* (18) |            |            |            |            |            |            |
| Locate               | 604* ( 5)  | 2594 ( 6)  | 3218* ( 6) | 3221 ( 6)  | 3223 ( 6)  | 1574* (13) | 1813* (13) | 1961* (13) |
|                      | 1429* (18) | 761* (19)  | 173 (20)   | 374* (20)  | 377 (20)   |            |            |            |
| LocatePageInterior   | 2375* (13) | 141* (24)  | 319* (36)  | 1092* (37) | 1895 (37)  |            |            |            |
| logicalSize          | 327* (27)  | 850* (28)  | 856 (28)   |            |            |            |            |            |
| LONGINT              | 21 ( 2)    | 199 ( 2)   | 51 ( 4)    | 536 ( 5)   | 599 ( 5)   | 633 ( 5)   | 634 ( 5)   | 635 ( 5)   |
|                      | 640 ( 5)   | 641 ( 5)   | 645 ( 5)   | 658 ( 5)   | 660 ( 5)   | 55 ( 6)    | 76 ( 6)    | 114 ( 6)   |
|                      | 265 ( 6)   | 308 ( 6)   | 395 ( 6)   | 395 ( 6)   | 423 ( 6)   | 423 ( 6)   | 476 ( 6)   | 476 ( 6)   |
|                      | 573 ( 6)   | 575 ( 6)   | 575 ( 6)   | 674 ( 6)   | 675 ( 6)   | 1129 ( 6)  | 1399 ( 6)  | 1441 ( 6)  |
|                      | 1622 ( 6)  | 1664 ( 6)  | 1843 ( 6)  | 1878 ( 6)  | 2035 ( 6)  | 2089 ( 6)  | 2231 ( 6)  | 2243 ( 6)  |
|                      | 2406 ( 6)  | 2475 ( 6)  | 2493 ( 6)  | 2678 ( 6)  | 2846 ( 6)  | 3175 ( 6)  | 3396 ( 6)  | 3476 ( 6)  |
|                      | 3480 ( 6)  | 3493 ( 6)  | 152 ( 7)   | 154 ( 7)   | 155 ( 7)   | 156 ( 7)   | 159 ( 7)   | 183 ( 7)   |
|                      | 189 ( 7)   | 193 ( 7)   | 206 ( 7)   | 19 ( 8)    | 26 ( 8)    | 74 ( 8)    | 141 ( 8)   | 145 ( 8)   |
|                      | 159 ( 8)   | 136 ( 9)   | 137 ( 9)   | 159 ( 9)   | 163 ( 9)   | 307 ( 9)   | 310 ( 9)   | 434 (10)   |
|                      | 436 (10)   | 841 (10)   | 847 (10)   | 890 (10)   | 892 (10)   | 901 (10)   | 903 (10)   | 938 (10)   |
|                      | 944 (10)   | 987 (10)   | 989 (10)   | 998 (10)   | 1000 (10)  | 1164 (10)  | 1248 (10)  | 1345 (10)  |
|                      | 1376 (10)  | 1518 (10)  | 1519 (10)  | 1520 (10)  | 1523 (10)  | 1569 (10)  | 1571 (10)  | 1583 (10)  |
|                      | 1585 (10)  | 1618 (10)  | 1619 (10)  | 1620 (10)  | 1623 (10)  | 1669 (10)  | 1671 (10)  | 1683 (10)  |
|                      | 1685 (10)  | 1788 (10)  | 1790 (10)  | 53 (11)    | 63 (11)    | 226 (11)   | 230 (11)   | 24 (12)    |
|                      | 80 (12)    | 81 (12)    | 108 (12)   | 151 (12)   | 153 (12)   | 155 (12)   | 187 (12)   | 188 (12)   |
|                      | 189 (12)   | 241 (12)   | 242 (12)   | 259 (12)   | 316 (12)   | 382 (12)   | 383 (12)   | 384 (12)   |
|                      | 385 (12)   | 424 (12)   | 513 (12)   | 540 (12)   | 552 (12)   | 553 (12)   | 554 (12)   | 555 (12)   |
|                      | 681 (12)   | 730 (12)   | 586 (13)   | 589 (13)   | 602 (13)   | 703 (13)   | 946 (13)   | 1028 (13)  |
|                      | 1065 (13)  | 1088 (13)  | 1107 (13)  | 1239 (13)  | 1665 (13)  | 1747 (13)  | 1748 (13)  | 2216 (13)  |
|                      | 2217 (13)  | 2218 (13)  | 2233 (13)  | 2286 (13)  | 2681 (13)  | 2694 (13)  | 2696 (13)  | 2700 (13)  |
|                      | 2703 (13)  | 2722 (13)  | 2832 (13)  | 2905 (13)  | 2931 (13)  | 2935 (13)  | 2936 (13)  | 2942 (13)  |
|                      | 2943 (13)  | 2948 (13)  | 2948 (13)  | 2951 (13)  | 2951 (13)  | 155 (14)   | 165 (14)   | 260 (14)   |
|                      | 260 (14)   | 300 (14)   | 731 (14)   | 783 (14)   | 843 (14)   | 910 (14)   | 985 (14)   | 1118 (14)  |
|                      | 117 (15)   | 162 (15)   | 162 (15)   | 164 (15)   | 164 (15)   | 323 (15)   | 193 (16)   | 199 (16)   |
|                      | 276 (16)   | 288 (16)   | 356 (16)   | 356 (16)   | 713 (16)   | 1090 (16)  | 1092 (16)  | 1119 (16)  |
|                      | 1220 (16)  | 1237 (16)  | 1389 (16)  | 1445 (16)  | 1465 (16)  | 1748 (16)  | 1948 (16)  | 1962 (16)  |
|                      | 2037 (16)  | 2140 (16)  | 2195 (16)  | 2252 (16)  | 2329 (16)  | 2440 (16)  | 2495 (16)  | 2946 (16)  |
|                      | 2957 (16)  | 19 (17)    | 308 (17)   | 352 (17)   | 367 (17)   | 411 (17)   | 582 (17)   | 636 (17)   |
|                      | 810 (17)   | 964 (17)   | 1045 (17)  | 1046 (17)  | 1047 (17)  | 1048 (17)  | 1049 (17)  | 1050 (17)  |
|                      | 1063 (17)  | 1064 (17)  | 1300 (17)  | 1314 (17)  | 1324 (17)  | 31 (18)    | 34 (18)    | 64 (18)    |
|                      | 75 (18)    | 75 (18)    | 87 (18)    | 87 (18)    | 97 (18)    | 97 (18)    | 100 (18)   | 102 (18)   |
|                      | 103 (18)   | 104 (18)   | 126 (18)   | 136 (18)   | 137 (18)   | 185 (18)   | 431 (18)   | 671 (18)   |
|                      | 958 (18)   | 958 (18)   | 966 (18)   | 966 (18)   | 1237 (18)  | 1239 (18)  | 1240 (18)  | 1835 (18)  |
|                      | 27 (19)    | 44 (19)    | 96 (19)    | 317 (19)   | 512 (19)   | 851 (19)   | 214 (20)   | 237 (21)   |
|                      | 631 (21)   | 643 (21)   | 943 (21)   | 1207 (21)  | 1213 (21)  | 1249 (21)  | 1345 (21)  | 1407 (21)  |
|                      | 155 (23)   | 156 (23)   | 165 (23)   | 177 (23)   | 134 (25)   | 156 (25)   | 212 (25)   | 213 (25)   |
|                      | 214 (25)   | 278 (25)   | 292 (25)   | 297 (25)   | 297 (25)   | 300 (25)   | 300 (25)   | 303 (25)   |
|                      | 306 (25)   | 310 (25)   | 315 (25)   | 323 (25)   | 323 (25)   | 390 (25)   | 402 (25)   | 409 (25)   |
|                      | 409 (25)   | 409 (25)   | 430 (25)   | 434 (25)   | 455 (25)   | 473 (25)   | 496 (25)   | 498 (25)   |
|                      | 172 (26)   | 204 (26)   | 297 (26)   | 331 (26)   | 389 (26)   | 479 (26)   | 500 (26)   | 500 (26)   |
|                      | 514 (26)   | 514 (26)   | 526 (26)   | 533 (26)   | 549 (26)   | 569 (26)   | 569 (26)   | 569 (26)   |
|                      | 674 (26)   | 682 (26)   | 738 (26)   | 738 (26)   | 785 (26)   | 876 (26)   | 1194 (26)  | 1208 (26)  |
|                      | 1321 (26)  | 1364 (26)  | 1376 (26)  | 160 (27)   | 228 (27)   | 304 (27)   | 357 (27)   | 361 (27)   |
|                      | 42 (28)    | 54 (28)    | 55 (28)    | 138 (28)   | 148 (28)   | 153 (28)   | 223 (28)   | 301 (28)   |
|                      | 326 (28)   | 330 (28)   | 405 (28)   | 413 (28)   | 571 (28)   | 574 (28)   | 615 (28)   | 617 (28)   |
|                      | 620 (28)   | 746 (28)   | 937 (28)   | 1036 (28)  | 1042 (28)  | 1043 (28)  | 1113 (28)  | 1115 (28)  |
|                      | 469 (30)   | 147 (31)   | 147 (31)   | 153 (31)   | 153 (31)   | 11 (33)    | 12 (33)    | 61 (33)    |
|                      | 61 (33)    | 135 (33)   | 135 (33)   | 47 (32)    | 221 (32)   | 65 (34)    | 20 (35)    | 28 (35)    |
|                      | 30 (35)    | 143 (36)   | 253 (37)   | 631 (37)   | 1131 (37)  | 1458 (37)  | 1531 (37)  | 1587 (37)  |
|                      | 1710 (37)  | 2073 (37)  | 2099 (37)  | 40 (38)    | 47 (38)    | 47 (38)    | 259 (38)   | 317 (38)   |
|                      | 396 (38)   | 104 (39)   | 105 (39)   | 106 (39)   | 107 (39)   | 270 (40)   | 833 (40)   | 1076 (40)  |
|                      | 1080 (40)  | 1089 (40)  | 1209 (40)  | 1464 (40)  | 22 (41)    | 123 (41)   | 216 (41)   | 357 (41)   |

|                           |            |            |            |            |            |            |            |            |
|---------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                           | 438 (41)   | 447 (41)   | 639 (41)   | 708 (41)   | 709 (41)   | 824 (41)   | 912 (41)   |            |
| LongInt                   | 178 (23)   | 195 (30)   |            |            |            |            |            |            |
| LongOffset                | 873* (18)  | 896 (18)   | 896 (18)   | 921- (18)  |            |            |            |            |
| LookForScrapType          | 175* (23)  | 187- (23)  | 201 (23)   | 202 (23)   |            |            |            |            |
| LookupErrString           | 2910* (13) | 198 (14)   | 208 (14)   | 215 (14)   | 714* (14)  | 772- (14)  | 774- (14)  |            |
| LookupObjName             | 133* (31)  | 77 (33)    | 241* (32)  | 278 (32)   |            |            |            |            |
| LookupSymbol              | 602* (13)  | 2832* (13) | 495 (14)   | 783* (14)  | 786- (14)  | 786 (14)   | 788- (14)  | 323* (15)  |
|                           | 326- (15)  | 326 (15)   | 328- (15)  |            |            |            |            |            |
| low                       | 682* (12)  | 693- (12)  | 696 (12)   | 701- (12)  | 702 (12)   | 731* (12)  | 750- (12)  | 752 (12)   |
|                           | 754 (12)   | 765- (12)  | 104* (30)  | 115- (30)  | 120 (30)   | 122 (30)   | 133- (30)  |            |
| lowd                      | 166* (14)  | 189 (14)   | 193 (14)   | 198 (14)   | 202 (14)   | 211 (14)   |            |            |
| lowErr                    | 726* (14)  | 753 (14)   | 755 (14)   |            |            |            |            |            |
| lowWord                   | 2273 ( 6)  | 2605 (16)  |            |            |            |            |            |            |
| lowWrd                    | 1973 (16)  | 858 (19)   | 958 (26)   |            |            |            |            |            |
| lowSpaceRes               | 727* (28)  | 730 (28)   | 734 (28)   |            |            |            |            |            |
| lowSpaceReserve           | 304* (27)  | 326* (28)  | 360- (28)  | 362- (28)  | 374 (28)   | 381- (28)  |            |            |
| lowSpaceVal               | 404* (28)  | 497 (28)   |            |            |            |            |            |            |
| lowWord                   | 183* ( 7)  |            |            |            |            |            |            |            |
| lto                       | 467* (18)  | 471 (18)   | 472 (18)   |            |            |            |            |            |
| lv                        | 467* (18)  | 469 (18)   | 472 (18)   |            |            |            |            |            |
| -M-                       |            |            |            |            |            |            |            |            |
| m                         | 189* ( 7)  | 19* ( 8)   | 335* (26)  | 339 (26)   | 342 (26)   | 344 (26)   | 167* (33)  | 176 (33)   |
|                           | 178 (33)   | 179 (33)   | 182 (33)   | 184 (33)   |            |            |            |            |
| MacAppAlert               | 2918* (13) | 797* (14)  | 804- (14)  | 1235 (14)  | 168 (17)   | 333 (17)   | 773 (17)   | 1158 (17)  |
| machineType               | 77 ( 4)    | 127 ( 4)   | 175 ( 4)   | 208 ( 4)   | 173* (25)  | 115 (26)   |            |            |
| MacPrint                  | 19* (36)   |            |            |            |            |            |            |            |
| MacsBug                   | 439* (25)  |            |            |            |            |            |            |            |
| MacsBugStr                | 442* (25)  |            |            |            |            |            |            |            |
| MacsbugStr                | 236 (28)   |            |            |            |            |            |            |            |
| MagentaBit                | 73* ( 6)   | 98 ( 6)    | 294* (26)  | 319 (26)   |            |            |            |            |
| magentaColor              | 133 ( 6)   | 804 (26)   |            |            |            |            |            |            |
| MAGetNewMBar              | 9* (16)    | 26- (16)   | 104 (16)   | 128 (16)   |            |            |            |            |
| MainEventLoop             | 712* (13)  | 1902* (16) | 2711 (16)  |            |            |            |            |            |
| MAInsertMenu              | 2260 ( 6)  | 2303 ( 6)  | 2510 ( 6)  | 231* (29)  | 355* (30)  |            |            |            |
| MakeClipboardWindow       | 1035* (13) | 150 (16)   | 1921* (16) | 1925- (16) | 1927- (16) |            |            |            |
| MakeFirstSubview          | 1532* (13) | 1460* (18) |            |            |            |            |            |            |
| MakeInspector             | 59* (14)   | 84 (16)    |            |            |            |            |            |            |
| MakeInspectorWindow       | 60* (14)   | 795 (16)   |            |            |            |            |            |            |
| MakeLastSubview           | 1534* (13) | 1475* (18) |            |            |            |            |            |            |
| MakeNewCopy               | 1265* (13) | 615* (17)  | 1272 (17)  | 1352 (17)  |            |            |            |            |
| MakeNewRgn                | 104 (10)   | 107 (10)   | 110 (10)   | 113 (10)   | 116 (10)   | 204 (10)   | 207 (10)   | 210 (10)   |
|                           | 213 (10)   | 216 (10)   | 2794 (10)  | 2796 (10)  | 2971* (13) | 568 (14)   | 569 (14)   | 579 (14)   |
|                           | 814* (14)  | 820- (14)  | 443 (18)   | 444 (18)   | 922 (18)   | 1579 (18)  |            |            |
| makeResident              | 248* (27)  | 1013* (28) | 1016 (28)  |            |            |            |            |            |
| MakeTERecord              | 173* (38)  | 51 (40)    | 107 (40)   | 1202* (40) |            |            |            |            |
| MakeTEView                | 110* ( 5)  | 286 ( 6)   | 331 ( 6)   | 611* ( 6)  | 625- ( 6)  |            |            |            |
| MakeViewForAlienClipboard | 1044* (13) | 1935* (16) | 1941- (16) | 2507 (16)  |            |            |            |            |
| makingCopy                | 1181* (13) | 1232* (13) | 1265* (13) | 1285* (13) | 1293* (13) | 1302* (13) | 1317* (13) | 1333* (13) |
|                           | 137* (17)  | 410* (17)  | 615* (17)  | 686 (17)   | 707 (17)   | 902* (17)  | 940 (17)   | 1037* (17) |
|                           | 1073 (17)  | 1087 (17)  | 1098 (17)  | 1103 (17)  | 1105 (17)  | 1133 (17)  | 1173 (17)  | 1195 (17)  |
|                           | 1206* (17) | 1248* (17) | 1272 (17)  | 1288* (17) | 1331 (17)  | 1352 (17)  | 1357 (17)  |            |
|                           | 952* (40)  | 958- (40)  | 964 (40)   | 973 (40)   | 974 (40)   |            |            |            |
| manyChars                 | 1146* (26) | 1161 (26)  |            |            |            |            |            |            |
| mapFalse                  | 124 (16)   | 1996 (16)  | 1998 (16)  | 2858 (16)  | 134* (29)  | 438 (30)   | 473 (30)   |            |
| mapTrue                   | 1145* (26) | 1159 (26)  |            |            |            |            |            |            |
| MatchesValue              | 167* ( 2)  | 170- ( 2)  | 174 ( 2)   | 244* ( 2)  | 247- ( 2)  | 251 ( 2)   | 303* ( 2)  | 306- ( 2)  |
|                           | 310 ( 2)   |            |            |            |            |            |            |            |
| matchState                | 194* ( 5)  | 234* ( 5)  | 974* ( 6)  | 976 ( 6)   | 1103* ( 6) | 1105 ( 6)  |            |            |
| MAX                       | 2198 ( 6)  | 2516 ( 6)  | 3499 ( 6)  | 2892 (10)  | 2893 (10)  | 2899 (10)  | 2900 (10)  | 122 (18)   |
|                           | 485 (37)   | 113 (39)   | 113 (39)   | 113 (39)   | 153 (39)   | 58 (40)    | 113 (40)   | 526 (40)   |
|                           | 531 (40)   | 683 (40)   | 113 (41)   | 195 (41)   | 501 (41)   | 685 (41)   | 796 (41)   |            |
| Max                       | 2046 ( 6)  | 2102 ( 6)  | 2592 ( 6)  | 876 (10)   | 876 (10)   | 973 (10)   | 973 (10)   | 1092 (10)  |
|                           | 1093 (10)  | 1603 (10)  | 1603 (10)  | 1703 (10)  | 1703 (10)  | 1867 (10)  | 1869 (10)  | 196 (20)   |
|                           | 263 (20)   | 614 (20)   | 1038 (21)  | 1145 (21)  | 297* (25)  | 500* (26)  | 503- (26)  | 505- (26)  |
|                           | 1471 (37)  | 1780 (37)  | 1781 (37)  | 208 (39)   | 210 (39)   | 621 (40)   | 1561 (40)  |            |
| MaxAddress                | 230* (11)  | 193 (12)   | 390 (12)   | 540* (12)  | 542- (12)  | 561 (12)   |            |            |
| MaxApplZone               | 505 (28)   |            |            |            |            |            |            |            |
| maxChars                  | 557* ( 5)  | 2959 ( 6)  | 2986- ( 6) |            |            |            |            |            |
| maxCoord                  | 605* (20)  | 614- (20)  | 615 (20)   | 617 (20)   | 619 (20)   | 621 (20)   | 622 (20)   |            |
| maximum                   | 635* ( 5)  | 3428 ( 6)  | 3456- ( 6) | 2218* (13) | 973 (21)   | 1006- (21) |            |            |
| MAXINT                    | 531 ( 6)   | 2601 ( 6)  | 3248 ( 6)  | 1115 (21)  | 328 (30)   | 70 (36)    | 29 (37)    | 277 (37)   |
|                           | 277 (37)   | 596 (37)   | 1496 (37)  | 55 (38)    | 314 (40)   | 319 (40)   | 319 (40)   | 443 (40)   |
|                           | 443 (40)   | 1128 (40)  | 1137 (40)  | 1605 (40)  | 1607 (40)  | 391 (41)   |            |            |
| MAXLONGINT                | 10 (14)    |            |            |            |            |            |            |            |
| maxName                   | 574* (17)  | 576 (17)   |            |            |            |            |            |            |
| maxNumber                 | 575* (17)  | 576 (17)   |            |            |            |            |            |            |
| maxP                      | 188* (12)  | 193- (12)  | 196 (12)   | 199 (12)   | 384* (12)  | 390- (12)  | 393 (12)   | 554* (12)  |
|                           | 561- (12)  | 564 (12)   |            |            |            |            |            |            |
| MaxPageNumber             | 2409* (13) | 150* (24)  | 152- (24)  | 279* (36)  | 1107* (37) | 1110- (37) | 1472 (37)  |            |
| maxPrefix                 | 576* (17)  | 591 (17)   | 592 (17)   |            |            |            |            |            |
| maxSize                   | 2022* (13) | 866* (19)  | 877 (19)   | 882 (19)   | 883 (19)   |            |            |            |
| mbar                      | 71* (16)   | 104- (16)  | 105 (16)   | 107 (16)   | 108 (16)   | 128- (16)  | 129 (16)   |            |
| MBarHeight                | 469* (14)  | 616 (14)   |            |            |            |            |            |            |
| MButtonHit                | 786 ( 6)   | 804 ( 6)   | 381* (13)  |            |            |            |            |            |
| mCancelHit                | 380* (13)  |            |            |            |            |            |            |            |
| mce                       | 150* ( 6)  | 175- ( 6)  | 176 ( 6)   | 179 ( 6)   | 184- ( 6)  | 185 ( 6)   | 186 ( 6)   | 211- ( 6)  |
|                           | 212 ( 6)   | 213 ( 6)   |            |            |            |            |            |            |
| MCEntry                   | 151 ( 6)   | 152 ( 6)   | 14 (16)    | 30 (16)    | 359 (30)   |            |            |            |
| MCEntryPtr                | 150 ( 6)   |            |            |            |            |            |            |            |
| mCheckBoxHit              | 868 ( 6)   | 886 ( 6)   | 938 ( 6)   | 382* (13)  |            |            |            |            |
| mClusterHit               | 1148 ( 6)  | 1163 ( 6)  | 383* (13)  |            |            |            |            |            |
| mControlHit               | 398* (13)  |            |            |            |            |            |            |            |
| MCTableHandle             | 20 (16)    |            |            |            |            |            |            |            |
| mctRGB1                   | 199 ( 6)   | 204 ( 6)   |            |            |            |            |            |            |
| mctRGB2                   | 200 ( 6)   | 213 ( 6)   | 222 ( 6)   |            |            |            |            |            |
| mctRGB3                   | 215 ( 6)   | 217 ( 6)   |            |            |            |            |            |            |
| mctRGB4                   | 205 ( 6)   | 220 ( 6)   |            |            |            |            |            |            |
| mDebug                    | 119 (16)   | 139* (29)  | 206 (30)   | 210 (30)   |            |            |            |            |

[illegible]



|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| modifiers           | 1193-(16) | 1312 (16) | 1318 (16) | 1398 (16) | 1399 (16) | 1400 (16) | 1401 (16) | 1402 (16) |  |
| mods                | 1403 (16) | 1587 (16) | 2558 (16) |           |           |           |           |           |  |
|                     | 2537*(16) | 2543-(16) | 2546-(16) | 2548-(16) | 2550-(16) | 2552-(16) | 2554-(16) | 2556-(16) |  |
|                     | 2559-(16) | 2561-(16) | 2562 (16) |           |           |           |           |           |  |
| mOKHit              | 379*(13)  | 107 (21)  | 140 (21)  |           |           |           |           |           |  |
| monaco              | 441 (13)  |           |           |           |           |           |           |           |  |
| moreMast            | 362 (14)  | 363-(14)  | 365-(14)  |           |           |           |           |           |  |
| MoreMasters         | 364 (14)  |           |           |           |           |           |           |           |  |
| mouseDidMove        | 654*( 9)  | 682*( 9)  | 687*( 9)  | 692*( 9)  | 2821*(10) | 2870*(10) | 2931*(10) | 2946 (10) |  |
|                     | 2993 (10) | 3030*(10) | 2129*(13) | 2133*(13) | 2487*(13) | 2494*(13) | 2536*(13) | 2539*(13) |  |
|                     | 3189*(16) | 3195 (16) | 66*(21)   | 68 (21)   | 77*(21)   | 80 (21)   | 461*(21)  | 471*(21)  |  |
|                     | 82*(22)   | 85 (22)   | 99*(22)   |           |           |           |           |           |  |
| mouseDown           | 639 (16)  | 2567 (16) | 2569 (16) |           |           |           |           |           |  |
| mouseUp             | 636 (16)  | 2567 (16) |           |           |           |           |           |           |  |
| MoveByUser          | 1963*(13) | 1621 (16) | 784*(19)  |           |           |           |           |           |  |
| MoveControl         | 784 (21)  |           |           |           |           |           |           |           |  |
| movedOnce           | 3110*(16) | 3269-(16) | 3280 (16) | 3286-(16) | 3289 (16) | 3331 (16) |           |           |  |
| MoveHil             | 1120 (14) | 131 (15)  | 437 (16)  | 1256 (16) | 49 (18)   | 124 (18)  | 158 (28)  | 923 (28)  |  |
|                     | 1147 (40) | 137 (41)  | 226 (41)  |           |           |           |           |           |  |
| MoveL               | 19*(35)   | 29*(35)   | 158-(35)  | 193-(35)  |           |           |           |           |  |
| MoveParameter       | 28*(35)   | 192-(35)  |           |           |           |           |           |           |  |
| MoveTo              | 2402 ( 6) | 2408 ( 6) | 2323 (10) | 469 (18)  | 372 (37)  | 791 (37)  | 806 (37)  | 810 (37)  |  |
|                     | 814 (37)  | 818 (37)  |           |           |           |           |           |           |  |
| MoveWindow          | 366 (19)  | 616 (19)  | 765 (19)  | 991 (19)  |           |           |           |           |  |
| mPatternHit         | 1646 ( 6) | 1691 ( 6) | 397*(13)  |           |           |           |           |           |  |
| mPictureHit         | 1860 ( 6) | 1895 ( 6) | 388*(13)  |           |           |           |           |           |  |
| mPopupHit           | 2062 ( 6) | 2119 ( 6) | 389*(13)  |           |           |           |           |           |  |
| mRadioHit           | 996 ( 6)  | 1014 ( 6) | 1065 ( 6) | 1235 ( 6) | 390*(13)  |           |           |           |  |
| mSCSIPort           | 83*(26)   | 114 (26)  |           |           |           |           |           |           |  |
| msePt               | 178*(39)  | 189 (39)  | 197 (39)  |           |           |           |           |           |  |
| msgAlert            | 247*(13)  | 158 (14)  |           |           |           |           |           |           |  |
| msgAltRecovery      | 256*(13)  | 160 (14)  |           |           |           |           |           |           |  |
| msgCancelled        | 263*(13)  | 169 (17)  | 202 (17)  | 949 (17)  | 1161 (17) |           |           |           |  |
| msgCmdErr           | 243*(13)  | 157 (14)  | 1967 (16) | 2266 (16) |           |           |           |           |  |
| msgDrawFailed       | 273*(13)  |           |           |           |           |           |           |           |  |
| msgExportClipFailed | 275*(13)  | 205 (16)  |           |           |           |           |           |           |  |
| msgImportClipFailed | 274*(13)  | 396 (16)  | 2500 (16) |           |           |           |           |           |  |
| msgInitFailed       | 265*(13)  | 307 (14)  |           |           |           |           |           |           |  |
| msgLookup           | 250*(13)  | 159 (14)  |           |           |           |           |           |           |  |
| msgNewFailed        | 269*(13)  | 2143 (16) |           |           |           |           |           |           |  |
| msgOpenFailed       | 270*(13)  | 1455 (16) | 2050 (16) | 2203 (16) |           |           |           |           |  |
| msgPrintFailed      | 268*(13)  | 1453 (16) | 1536 (37) | 1539 (37) |           |           |           |           |  |
| msgRevertFailed     | 267*(13)  | 970 (17)  |           |           |           |           |           |           |  |
| msgSaveAsFailed     | 271*(13)  | 1076 (17) |           |           |           |           |           |           |  |
| msgSaveCopyFailed   | 272*(13)  | 1074 (17) |           |           |           |           |           |           |  |
| msgSaveFailed       | 266*(13)  | 1072 (17) |           |           |           |           |           |           |  |
| msgSent             | 503*(37)  | 569-(37)  | 574-(37)  | 577 (37)  |           |           |           |           |  |
| msgStrList          | 261*(13)  | 263 (13)  | 265 (13)  | 266 (13)  | 267 (13)  | 268 (13)  | 269 (13)  | 270 (13)  |  |
|                     | 271 (13)  | 272 (13)  | 273 (13)  | 274 (13)  | 275 (13)  |           |           |           |  |
| mStaticTextHit      | 2698 ( 6) | 2714 ( 6) | 391*(13)  |           |           |           |           |           |  |
| Munger              | 681 ( 6)  | 891 (10)  | 902 (10)  | 988 (10)  | 999 (10)  | 1570 (10) | 1584 (10) | 1670 (10) |  |
|                     | 1684 (10) | 207 (12)  | 455 (12)  | 140 (28)  | 939 (28)  | 85 (32)   | 777 (41)  |           |  |
| mUpMask             | 3336 (16) | 54 (37)   |           |           |           |           |           |           |  |
| mustAdaptToScreen   | 1876*(13) | 151 (19)  | 196-(19)  |           |           |           |           |           |  |
| mustForceOnScreen   | 1878*(13) | 157 (19)  | 198-(19)  |           |           |           |           |           |  |
| mVScrollBarHit      | 393*(13)  | 952 (21)  |           |           |           |           |           |           |  |
| myDlgMask           | 54*(37)   | 67 (37)   |           |           |           |           |           |           |  |
| myExtent            | 3316*( 6) | 3333*( 6) | 3344 ( 6) | 3345 ( 6) | 3349 ( 6) | 186*(20)  | 193 (20)  | 195 (20)  |  |
|                     | 197 (20)  | 432*(20)  | 439 (20)  | 443 (20)  | 444 (20)  | 446 (20)  | 482*(20)  | 489 (20)  |  |
|                     | 490 (20)  |           |           |           |           |           |           |           |  |
| myFrame             | 1007*(18) | 1028 (18) | 1029 (18) | 1032 (18) | 1556*(18) | 1571 (18) | 1580 (18) | 1584 (18) |  |
|                     | 1585 (18) |           |           |           |           |           |           |           |  |
| myType              | 1093*(16) |           |           |           |           |           |           |           |  |
| -N-                 |           |           |           |           |           |           |           |           |  |
| n                   | 67*(33)   | 77 (33)   | 79 (33)   | 164*(33)  | 174-(33)  | 186 (33)  |           |           |  |
| name                | 1192*(13) | 334*(14)  | 158*(17)  | 165-(17)  | 167 (17)  | 300*(17)  | 331-(17)  | 332 (17)  |  |
|                     | 720*(17)  | 725 (17)  | 758*(17)  | 770-(17)  | 771 (17)  | 1040*(17) | 1069 (17) | 1087 (17) |  |
|                     | 1090*(17) | 1103 (17) | 1133 (17) | 1144 (17) | 1157 (17) | 1173 (17) | 1181 (17) | 1196 (17) |  |
|                     | 390*(25)  | 406*(25)  | 413*(25)  | 172*(26)  | 176 (26)  | 233*(26)  | 238 (26)  | 579*(26)  |  |
|                     | 604 (26)  | 640 (26)  | 653 (26)  | 910*(26)  | 915 (26)  | 917 (26)  | 923*(26)  | 929 (26)  |  |
|                     | 931 (26)  | 802*(28)  | 816 (28)  | 817 (28)  | 819 (28)  | 822 (28)  |           |           |  |
| nameList            | 799*(28)  | 809-(28)  | 810 (28)  | 813 (28)  | 815 (28)  | 827 (28)  | 828 (28)  |           |  |
| namePtr             | 386*(25)  | 133*(26)  | 139 (26)  |           |           |           |           |           |  |
| nameSelProcs        | 328*(14)  |           |           |           |           |           |           |           |  |
| nChars              | 833*(40)  | 846-(40)  | 847 (40)  | 853 (40)  | 217*(41)  | 220-(41)  | 221 (41)  | 230 (41)  |  |
|                     | 233 (41)  |           |           |           |           |           |           |           |  |
| NeedAdjust          | 371*(40)  | 373-(40)  | 380 (40)  | 382 (40)  |           |           |           |           |  |
| NeedCalcMenuSize    | 236*(29)  | 390*(30)  |           |           |           |           |           |           |  |
| needCalText         | 1260*(40) | 1272-(40) | 1274 (40) |           |           |           |           |           |  |
| needed              | 54*(28)   | 225*(28)  | 247-(28)  | 250 (28)  | 260 (28)  | 261 (28)  | 264 (28)  | 267 (28)  |  |
|                     | 273-(28)  | 276 (28)  | 615*(28)  |           |           |           |           |           |  |
| neededBlks          | 1047*(17) | 1129-(17) | 1131 (17) | 1152 (17) | 1185 (17) | 1188 (17) |           |           |  |
| needNewCommand      | 698*(40)  | 736-(40)  | 737 (40)  | 738-(40)  | 740 (40)  |           |           |           |  |
| needsResizing       | 1711*(18) | 1715-(18) | 1719-(18) | 1723-(18) | 1726 (18) |           |           |           |  |
| networkEvt          | 2606 (16) |           |           |           |           |           |           |           |  |
| NEW                 | 31 ( 6)   | 32 ( 6)   | 33 ( 6)   | 34 ( 6)   | 35 ( 6)   | 36 ( 6)   | 37 ( 6)   | 38 ( 6)   |  |
|                     | 39 ( 6)   | 40 ( 6)   | 41 ( 6)   | 42 ( 6)   | 43 ( 6)   | 44 ( 6)   | 276 ( 6)  | 323 ( 6)  |  |
|                     | 617 ( 6)  | 640 (10)  | 649 (10)  | 658 (10)  | 667 (10)  | 692 (14)  | 693 (14)  | 694 (14)  |  |
|                     | 695 (14)  | 696 (14)  | 307 (21)  | 27 (39)   |           |           |           |           |  |
| New                 | 97 ( 2)   | 226 ( 2)  | 41 (12)   | 645 (14)  | 865 (14)  | 938 (14)  | 1003 (14) | 146 (16)  |  |
|                     | 227 (20)  | 239 (20)  | 199 (37)  | 1343 (37) | 767 (40)  | 774 (40)  | 781 (40)  | 801 (40)  |  |
|                     | 817 (40)  | 371 (41)  |           |           |           |           |           |           |  |
| newAutoWrap         | 273*(38)  | 347*(40)  | 349 (40)  | 351 (40)  |           |           |           |           |  |
| newBkColor          | 2251*( 6) | 2263 ( 6) | 2264 ( 6) | 2274 ( 6) | 2298*( 6) | 2313 ( 6) | 2314 ( 6) | 2318 ( 6) |  |
|                     | 2319 ( 6) |           |           |           |           |           |           |           |  |
| newBRRect           | 2878*(10) |           |           |           |           |           |           |           |  |
| newChoice           | 2242*( 6) | 2273-( 6) | 2279 ( 6) |           |           |           |           |           |  |
| newCommand          | 1953*(16) |           |           |           |           |           |           |           |  |

|                   |            |            |            |            |            |            |            |            |  |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| NewControl        | 641 (21)   |            |            |            |            |            |            |            |  |
| NewCWindow        | 120 (19)   |            |            |            |            |            |            |            |  |
| newDelay          | 79* (3)    | 88* (4)    | 93 (4)     | 94 (4)     | 97- (4)    | 99 (4)     | 100 (4)    |            |  |
| newDocTitle       | 2006* (13) | 905* (19)  | 915 (19)   |            |            |            |            |            |  |
| newDownPt         | 622* (13)  | 262* (15)  | 265 (15)   | 267 (15)   | 268 (15)   |            |            |            |  |
| newError          | 1324* (17) |            |            |            |            |            |            |            |  |
| newFColor         | 2250* (6)  | 2263 (6)   | 2264 (6)   | 2274 (6)   | 2297* (6)  | 2313 (6)   | 2314 (6)   | 2318 (6)   |  |
|                   | 2319 (6)   |            |            |            |            |            |            |            |  |
| newFlags          | 469* (30)  | 475- (30)  | 477 (30)   | 479- (30)  | 479 (30)   | 480 (30)   | 484 (30)   |            |  |
| NewHandle         | 346 (16)   | 424 (16)   | 48 (18)    | 419 (28)   | 421 (28)   | 444 (28)   | 464 (28)   | 472 (28)   |  |
|                   | 856 (28)   | 91 (33)    | 50 (32)    |            |            |            |            |            |  |
| newHeight         | 2190* (6)  | 2198- (6)  | 2199 (6)   |            |            |            |            |            |  |
| newInset          | 2114* (13) | 438* (21)  | 442 (21)   |            |            |            |            |            |  |
| newInterior       | 1724* (13) | 1508* (18) | 1625* (37) | 1644- (37) | 1648 (37)  | 1652 (37)  | 1653 (37)  | 1661 (37)  |  |
| newItem           | 179* (11)  | 102* (12)  | 123 (12)   | 125 (12)   | 133 (12)   |            |            |            |  |
| newJust           | 301* (38)  | 1348* (40) | 1350 (40)  | 1351 (40)  |            |            |            |            |  |
| newKeyScript      | 326* (25)  | 748* (26)  | 755 (26)   | 756 (26)   |            |            |            |            |  |
| newLen            | 2369* (6)  | 2385- (6)  | 2389 (6)   | 2390- (6)  | 2390 (6)   | 2393- (6)  | 2393 (6)   | 2394 (6)   |  |
|                   | 2395 (6)   |            |            |            |            |            |            |            |  |
| newLength         | 437* (41)  | 474- (41)  | 476 (41)   | 486 (41)   | 487 (41)   |            |            |            |  |
| newLimit          | 1043* (28) | 1046- (28) | 1048 (28)  | 1049 (28)  |            |            |            |            |  |
| NewList           | 303* (11)  | 35* (12)   | 45- (12)   | 651 (14)   | 90 (16)    | 83 (17)    | 88 (17)    | 344 (18)   |  |
|                   | 1225 (21)  | 1259 (21)  |            |            |            |            |            |            |  |
| newLoc            | 413* (37)  | 417- (37)  | 420 (37)   | 423 (37)   | 428 (37)   | 1296* (40) |            |            |  |
| newMargins        | 365* (36)  | 1046* (37) | 1065 (37)  |            |            |            |            |            |  |
| newMenuState      | 2803* (16) | 2812- (16) | 2814- (16) | 2816- (16) | 2818 (16)  |            |            |            |  |
| newMessage        | 206* (7)   | 74* (8)    | 77 (8)     | 1324* (17) |            |            |            |            |  |
| newMsg            | 1064* (17) | 1072- (17) | 1074- (17) | 1076- (17) | 1078 (17)  |            |            |            |  |
| newName           | 1231* (13) | 136* (17)  |            |            |            |            |            |            |  |
| NewPaletteWindow  | 2851* (13) | 830* (14)  | 886- (14)  |            |            |            |            |            |  |
| NewPermHandle     | 73 (10)    | 77 (10)    | 179 (10)   | 183 (10)   | 374 (17)   | 206 (23)   | 327* (27)  | 850* (28)  |  |
|                   | 856- (28)  | 340 (30)   | 89 (33)    | 1817 (37)  | 1947 (37)  | 2110 (37)  | 2114 (37)  | 1468 (40)  |  |
|                   | 55 (41)    | 63 (41)    | 464 (41)   | 468 (41)   | 652 (41)   |            |            |            |  |
|                   | 51 (33)    | 77 (35)    |            |            |            |            |            |            |  |
| NewPtr            | 2874* (10) |            |            |            |            |            |            |            |  |
| newQDRect         | 2873* (10) | 2890 (10)  | 2891 (10)  | 2892 (10)  | 2893 (10)  | 2897 (10)  | 2898 (10)  | 2899 (10)  |  |
| newRect           | 2900 (10)  | 2904 (10)  | 2920 (10)  |            |            |            |            |            |  |
|                   | 818 (14)   |            |            |            |            |            |            |            |  |
| NewRgn            | 274* (9)   | 488* (10)  | 502 (10)   | 502 (10)   | 508 (10)   | 526 (10)   | 531 (10)   |            |  |
| newSelLoc         | 1297* (40) |            |            |            |            |            |            |            |  |
| newSelRect        | 1298* (40) | 1307 (40)  | 1313 (40)  | 1317 (40)  | 1320 (40)  | 1321 (40)  | 1325 (40)  | 1330 (40)  |  |
| NewSimpleWindow   | 2860* (13) | 898* (14)  | 963- (14)  | 1927 (16)  |            |            |            |            |  |
| newSize           | 1564* (13) | 2089* (13) | 378* (18)  | 381- (18)  | 382 (18)   | 383 (18)   | 383 (18)   | 385 (18)   |  |
|                   | 385 (18)   | 619* (18)  | 638 (18)   | 661 (18)   | 1710* (18) | 1716- (18) | 1722 (18)  | 1722 (18)  |  |
|                   | 1728 (18)  | 1729 (18)  | 1729 (18)  | 308* (19)  | 320- (19)  | 321 (19)   | 322 (19)   | 323- (19)  |  |
|                   | 323 (19)   | 325 (19)   | 325 (19)   | 226* (21)  | 231 (21)   | 234 (21)   | 235 (21)   | 240* (38)  |  |
|                   | 366* (40)  | 378 (40)   | 381 (40)   | 381 (40)   | 383 (40)   | 383 (40)   |            |            |  |
| NewString         | 31 (2)     | 34 (2)     | 1354 (6)   | 2885 (6)   | 535 (14)   |            |            |            |  |
| newStuff          | 2883* (13) | 1242* (14) | 1248 (14)  | 1252 (14)  |            |            |            |            |  |
| newStyl           | 517* (38)  | 541* (41)  | 544 (41)   |            |            |            |            |            |  |
| newStyle          | 1366* (40) | 1385- (40) | 1388- (40) | 1391 (40)  | 1393 (40)  | 1396 (40)  | 1398 (40)  | 1400 (40)  |  |
|                   | 1400 (40)  | 1402 (40)  | 1407 (40)  | 1411 (40)  | 1420 (40)  |            |            |            |  |
| newStyleLen       | 438* (41)  | 460- (41)  | 491- (41)  | 492 (41)   | 496 (41)   |            |            |            |  |
| newStyles         | 40* (38)   | 440* (41)  | 451 (41)   | 461- (41)  | 468- (41)  | 469 (41)   | 491 (41)   | 494 (41)   |  |
| newStyleSize      | 106* (39)  | 118- (39)  | 150- (39)  | 153 (39)   |            |            |            |            |  |
| newStyls          | 520* (38)  | 551* (41)  | 556 (41)   |            |            |            |            |            |  |
| NewSysPtr         | 303* (25)  | 526* (26)  | 79 (35)    |            |            |            |            |            |  |
| newTarget         | 812* (13)  | 2024* (13) | 2787* (16) | 2791 (16)  | 2792 (16)  | 892* (19)  | 895 (19)   | 897 (19)   |  |
| NewTemplateWindow | 2871* (13) | 1020* (14) | 1040- (14) | 1925 (16)  |            |            |            |            |  |
| newTRect          | 330* (38)  | 1553* (40) | 1560 (40)  | 1564 (40)  | 1565 (40)  |            |            |            |  |
| newText           | 439* (41)  | 450 (41)   | 462- (41)  | 464- (41)  | 465 (41)   | 474 (41)   | 484 (41)   |            |  |
| newTitle          | 2134* (16) | 2155 (16)  | 2157 (16)  | 2158 (16)  | 2163 (16)  | 2164 (16)  | 2166 (16)  |            |  |
| newTracker        | 3173* (16) | 3175 (16)  | 3218* (16) | 3231- (16) | 3233 (16)  | 3236 (16)  | 3238 (16)  | 3238 (16)  |  |
|                   | 3239 (16)  |            |            |            |            |            |            |            |  |
| newTranslation    | 607* (20)  | 611- (20)  | 622 (20)   | 626 (20)   | 626 (20)   | 628 (20)   | 628 (20)   |            |  |
| NewTWindow        | 2865* (13) | 849 (14)   | 916 (14)   | 975* (14)  | 1009- (14) |            |            |            |  |
| newVal            | 2198* (13) | 852* (21)  | 859 (21)   |            |            |            |            |            |  |
| newValue          | 660* (5)   | 3493* (6)  | 3499- (6)  | 3499 (6)   | 3500 (6)   | 532* (20)  | 536- (20)  | 539- (20)  |  |
|                   | 542 (20)   | 544 (20)   |            |            |            |            |            |            |  |
| newView           | 61* (15)   | 82- (15)   | 83 (15)    | 84 (15)    | 85 (15)    |            |            |            |  |
| NewViewRsrc       | 2927* (13) | 41* (18)   | 57- (18)   |            |            |            |            |            |  |
| newVolRefnum      | 1232* (13) | 136* (17)  |            |            |            |            |            |            |  |
| newVRect          | 2876* (10) |            |            |            |            |            |            |            |  |
| newWid            | 2368* (6)  | 2382- (6)  | 2383 (6)   | 2389- (6)  | 2389 (6)   | 2391 (6)   |            |            |  |
| newWidth          | 2191* (6)  | 2197- (6)  | 2199 (6)   |            |            |            |            |            |  |
| NewWindow         | 123 (19)   |            |            |            |            |            |            |            |  |
| next              | 120* (5)   | 593* (6)   | 597 (6)    | 600- (6)   | 602 (6)    | 603 (6)    | 713* (6)   | 729 (6)    |  |
|                   | 730- (6)   | 737- (6)   | 742 (6)    | 743- (6)   |            |            |            |            |  |
|                   | 647 (21)   |            |            |            |            |            |            |            |  |
| nextControl       | 615* (13)  | 213* (15)  |            |            |            |            |            |            |  |
| nextEvent         | 888* (16)  | 895- (16)  | 897 (16)   | 1029* (16) | 1036- (16) | 1042 (16)  |            |            |  |
| nextHandler       | 157* (7)   | 171 (8)    | 311 (8)    |            |            |            |            |            |  |
| nextInfo          | 66* (34)   | 86- (35)   | 220 (35)   | 224 (35)   | 226 (35)   | 231- (35)  | 231 (35)   |            |  |
| nextPatch         | 653* (9)   | 686* (9)   | 691* (9)   | 2820* (10) | 2930* (10) | 2941 (10)  | 3029* (10) | 2123* (13) |  |
| nextPoint         | 2128* (13) | 2132* (13) | 2481* (13) | 2486* (13) | 2493* (13) | 2535* (13) | 2538* (13) | 2542* (13) |  |
|                   | 55* (21)   | 58 (21)    | 65* (21)   | 68 (21)    | 76* (21)   | 80 (21)    | 82 (21)    | 452* (21)  |  |
|                   | 460* (21)  | 470* (21)  | 477 (21)   | 482 (21)   | 72* (22)   | 81* (22)   | 87 (22)    | 98* (22)   |  |
| NGetTrapAddress   | 104 (26)   | 84 (35)    |            |            |            |            |            |            |  |
| nLines            | 103 (23)   | 68 (39)    | 313 (40)   | 331 (40)   | 460 (40)   | 471 (40)   | 611 (40)   | 620 (40)   |  |
| nMenus            | 64* (16)   | 136 (16)   |            |            |            |            |            |            |  |
| nmPtr             | 3444* (6)  | 3449- (6)  | 3452 (6)   |            |            |            |            |            |  |
| noErr             | 1875 (18)  | 1211 (37)  |            |            |            |            |            |            |  |
| noErr             | 1276 (6)   | 1290 (6)   | 1355 (6)   | 2788 (6)   | 2796 (6)   | 2886 (6)   | 44 (8)     | 53 (8)     |  |
|                   | 95 (8)     | 104 (8)    | 123 (8)    | 132 (8)    | 103 (14)   | 304 (14)   | 1125 (14)  | 295 (16)   |  |
|                   | 310 (16)   | 361 (16)   | 533 (16)   | 1362 (16)  | 1366 (16)  | 1447 (16)  | 1467 (16)  | 2090 (16)  |  |
|                   | 2101 (16)  | 2334 (16)  | 169 (17)   | 172 (17)   | 202 (17)   | 249 (17)   | 257 (17)   | 480 (17)   |  |
|                   | 535 (17)   | 552 (17)   | 641 (17)   | 647 (17)   | 815 (17)   | 945 (17)   | 949 (17)   | 1146 (17)  |  |
|                   | 1161 (17)  | 1182 (17)  | 1264 (17)  | 1305 (17)  | 1366 (17)  | 31 (26)    | 39 (26)    | 146 (26)   |  |
|                   | 176 (26)   | 225 (26)   | 244 (26)   | 593 (26)   | 660 (26)   | 666 (26)   | 462 (28)   | 208 (33)   |  |

|                       |           |           |           |           |           |           |           |           |
|-----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                       | 132 (35)  | 119 (37)  | 614 (37)  | 639 (37)  | 1977 (37) | 1099 (40) | 1120 (40) | 1129 (40) |
|                       | 1154 (40) |           |           |           |           |           |           |           |
| noName                | 1224*(13) | 1463*(17) | 1468 (17) | 1469 (17) | 1473 (17) |           |           |           |
| noOfBytes             | 303*(25)  | 526*(26)  |           |           |           |           |           |           |
| noOfCopies            | 1124*(37) | 1151*(37) | 1153*(37) | 1165 (37) |           |           |           |           |
| noOfDigits            | 293*(25)  | 480*(26)  | 485-(26)  | 485 (26)  | 486 (26)  | 487 (26)  |           |           |
| noOfHierMenus         | 194*(30)  | 225-(30)  | 226 (30)  |           |           |           |           |           |
| noOfMenus             | 192*(30)  | 200-(30)  | 202 (30)  |           |           |           |           |           |
| normalizedPageNum     | 1227*(37) | 1231-(37) | 1235 (37) | 1236 (37) |           |           |           |           |
| NotifySubView         | 538*(18)  | 546 (18)  | 1434*(18) | 1452 (18) | 1562*(18) | 1602 (18) |           |           |
| NOTpatXOR             | 1222 (14) |           |           |           |           |           |           |           |
| notUsed               | 184*(30)  |           |           |           |           |           |           |           |
| NotYetImplemented     | 2834*(13) | 1051*(14) |           |           |           |           |           |           |
| noTypeErr             | 184 (23)  | 1166 (40) |           |           |           |           |           |           |
| now                   | 2957*(16) | 2965-(16) | 2966 (16) | 2969 (16) |           |           |           |           |
| NSetTrapAddress       | 121 (35)  | 131 (35)  | 165 (35)  | 200 (35)  | 244 (35)  |           |           |           |
| nSlots                | 946*(19)  | 974-(19)  | 976-(19)  | 978 (19)  | 981 (19)  |           |           |           |
| nStrips               | 440*(37)  | 447-(37)  | 447 (37)  | 467-(37)  | 469 (37)  |           |           |           |
| nullEvent             | 2565 (16) |           |           |           |           |           |           |           |
| NullMenuProc          | 32*(30)   | 345 (30)  | 405*(30)  |           |           |           |           |           |
| NullStHandle          | 1489 (40) |           |           |           |           |           |           |           |
| nullStyle             | 1496 (40) | 1503-(40) |           |           |           |           |           |           |
| nullStyles            | 1489*(40) | 1496-(40) | 1503 (40) |           |           |           |           |           |
| Num                   | 435*(10)  | 463 (10)  | 465 (10)  | 846*(10)  | 877 (10)  | 881 (10)  | 883 (10)  | 883 (10)  |
|                       | 885 (10)  | 887 (10)  | 890 (10)  | 896 (10)  | 896 (10)  | 896 (10)  | 898 (10)  | 898 (10)  |
|                       | 899 (10)  | 901 (10)  | 909-(10)  | 909 (10)  | 943*(10)  | 974 (10)  | 978 (10)  | 980 (10)  |
|                       | 980 (10)  | 982 (10)  | 984 (10)  | 987 (10)  | 993 (10)  | 993 (10)  | 993 (10)  | 995 (10)  |
|                       | 995 (10)  | 996 (10)  | 998 (10)  | 1006-(10) | 1006 (10) | 1344*(10) | 1361 (10) | 1362 (10) |
|                       | 1375*(10) | 1392 (10) | 1393 (10) | 1522*(10) | 1550 (10) | 1551 (10) | 1553 (10) | 1553 (10) |
|                       | 1566-(10) | 1569 (10) | 1576 (10) | 1577 (10) | 1583 (10) | 1590 (10) | 1590 (10) | 1591 (10) |
|                       | 1591 (10) | 1592 (10) | 1593 (10) | 1594 (10) | 1622*(10) | 1650 (10) | 1651 (10) | 1653 (10) |
|                       | 1653 (10) | 1666-(10) | 1669 (10) | 1676 (10) | 1677 (10) | 1683 (10) | 1690 (10) | 1690 (10) |
|                       | 1691 (10) | 1691 (10) | 1692 (10) | 1693 (10) | 1694 (10) | 1789*(10) | 1817 (10) | 1819 (10) |
| num                   | 457-(10)  | 458 (10)  | 459 (10)  | 1811-(10) | 1812 (10) | 1813 (10) | 1466*(17) | 1471 (17) |
|                       | 1473 (17) |           |           |           |           |           |           |           |
| numberOfChars         | 3155*( 6) | 3163-( 6) | 3165 ( 6) | 3167 ( 6) |           |           |           |           |
| NumberTextTemplate    | 631 ( 5)  | 632*( 5)  | 3431 ( 6) | 3450 ( 6) |           |           |           |           |
| NumberTextTemplatePtr | 631*( 5)  | 3423 ( 6) | 3444 ( 6) | 3449 ( 6) |           |           |           |           |
| NumberToHex           | 306*(25)  | 533*(26)  | 948 (26)  | 950 (26)  | 952 (26)  | 954 (26)  | 979 (26)  | 1025 (26) |
|                       | 1026 (26) | 1028 (26) |           |           |           |           |           |           |
| NumBlocks             | 1129 (17) | 1129 (17) | 1149 (17) | 1150 (17) | 1184 (17) | 1184 (17) | 409*(25)  | 569*(26)  |
|                       | 572-(26)  |           |           |           |           |           |           |           |
| numBytes              | 409*(25)  | 569*(26)  | 572 (26)  | 361*(27)  | 1036*(28) | 1046 (28) |           |           |
| numEntries            | 13*(16)   | 43 (16)   | 358*(30)  | 378 (30)  |           |           |           |           |
| numItems              | 2189*( 6) | 2196-( 6) | 2198 ( 6) |           |           |           |           |           |
| numLines              | 35*(14)   | 442 (14)  |           |           |           |           |           |           |
| numOfCols             | 855 (10)  |           |           |           |           |           |           |           |
| numOfCols             | 94*( 9)   | 176*( 9)  | 269*( 9)  | 279*( 9)  | 284*( 9)  | 289*( 9)  | 325*( 9)  | 331*( 9)  |
|                       | 337*( 9)  | 362*( 9)  | 432*( 9)  | 34*(10)   | 126 (10)  | 127 (10)  | 224 (10)  | 225 (10)  |
|                       | 254-(10)  | 432*(10)  | 441 (10)  | 441 (10)  | 459 (10)  | 466 (10)  | 839*(10)  | 853 (10)  |
|                       | 853 (10)  | 871 (10)  | 879 (10)  | 915 (10)  | 1033*(10) | 1035 (10) | 1057*(10) | 1059 (10) |
|                       | 1059 (10) | 1515*(10) | 1527 (10) | 1529 (10) | 1546 (10) | 1553 (10) | 1577 (10) | 1594 (10) |
|                       | 1599 (10) | 1600 (10) | 1715*(10) | 1717 (10) | 1739*(10) | 1741 (10) | 1882*(10) | 1885 (10) |
|                       | 1885 (10) | 1896 (10) | 1897 (10) | 2204*(10) | 2217 (10) |           |           |           |
| numOfItems            | 526*( 9)  | 569*( 9)  | 570*( 9)  | 571*( 9)  | 591*( 9)  | 592*( 9)  | 593*( 9)  | 611*( 9)  |
|                       | 2445*(10) | 2457 (10) | 2520*(10) | 2522 (10) | 2533*(10) | 2535 (10) | 2546*(10) | 2548 (10) |
|                       | 2640*(10) | 2642 (10) | 2653*(10) | 2655 (10) | 2666*(10) | 2668 (10) | 2737*(10) | 2740 (10) |
| numOfRows             | 93*( 9)   | 175*( 9)  | 270*( 9)  | 280*( 9)  | 285*( 9)  | 290*( 9)  | 326*( 9)  | 332*( 9)  |
|                       | 338*( 9)  | 363*( 9)  | 431*( 9)  | 33*(10)   | 124 (10)  | 125 (10)  | 222 (10)  | 223 (10)  |
|                       | 253-(10)  | 936*(10)  | 950 (10)  | 950 (10)  | 952 (10)  | 967 (10)  | 976 (10)  | 1012 (10) |
|                       | 1045*(10) | 1047 (10) | 1068*(10) | 1070 (10) | 1070 (10) | 1615*(10) | 1627 (10) | 1629 (10) |
|                       | 1646 (10) | 1653 (10) | 1677 (10) | 1694 (10) | 1699 (10) | 1700 (10) | 1727*(10) | 1729 (10) |
|                       | 1751*(10) | 1753 (10) | 1786*(10) | 1795 (10) | 1795 (10) | 1813 (10) | 1820 (10) | 1908*(10) |
|                       | 1911 (10) | 1911 (10) | 1922 (10) | 1923 (10) | 2203*(10) | 2217 (10) |           |           |
| NumToString           | 3404 ( 6) | 3425 ( 6) | 3500 ( 6) | 637 (12)  | 596 (17)  | 1471 (17) | 944 (26)  | 946 (26)  |
|                       | 957 (26)  | 958 (26)  | 982 (26)  | 983 (26)  | 988 (26)  | 989 (26)  | 991 (26)  | 993 (26)  |
|                       | 999 (26)  | 1001 (26) | 1044 (26) | 1047 (26) | 1048 (26) | 1053 (26) | 1054 (26) | 1056 (26) |
|                       | 1058 (26) | 311 (28)  | 777 (28)  | 333 (30)  | 361 (37)  | 805 (37)  | 809 (37)  | 813 (37)  |
|                       | 817 (37)  |           |           |           |           |           |           |           |
| numTypes              | 340*(16)  | 351-(16)  | 352 (16)  | 353 (16)  | 421*(16)  | 428-(16)  | 430 (16)  | 432-(16)  |
|                       | 450 (16)  |           |           |           |           |           |           |           |
| numViews              | 1455*(13) | 108*(15)  | 136-(15)  | 136 (15)  | 141 (15)  | 54-(18)   | 267-(18)  | 267 (18)  |
| -O-                   |           |           |           |           |           |           |           |           |
| obj                   | 106*(31)  | 119*(31)  | 122*(31)  | 153*(31)  | 156*(31)  | 61*(33)   | 82-(33)   | 89-(33)   |
|                       | 91-(33)   | 94 (33)   | 99 (33)   | 108 (33)  | 111 (33)  | 123*(33)  | 125 (33)  | 137*(32)  |
|                       | 140 (32)  | 141 (32)  | 189*(32)  | 195 (32)  | 198 (32)  | 199 (32)  | 201 (32)  | 205 (32)  |
|                       | 219*(32)  | 223 (32)  | 226 (32)  |           |           |           |           |           |
| OBJFail               | 257 ( 8)  | 139*(31)  | 10*(33)   | 141 (33)  |           |           |           |           |
| objID                 | 50*(31)   | 133*(31)  | 82-(32)   | 118 (32)  | 241*(32)  | 243 (32)  |           |           |
| objName               | 51*(31)   | 133*(31)  | 63-(32)   | 69-(32)   | 116 (32)  | 241*(32)  | 243 (32)  |           |
| ObjNameRec            | 49*(31)   | 55 (31)   | 42 (32)   | 112 (32)  |           |           |           |           |
| objRec                | 42*(32)   | 63 (32)   | 69 (32)   | 82 (32)   | 86 (32)   | 86 (32)   |           |           |
| ODD                   | 57 (10)   | 62 (10)   | 157 (10)  | 162 (10)  | 2622 (16) | 2860 (16) | 79 (18)   | 110 (18)  |
|                       | 1111 (18) | 652 (19)  | 419 (26)  | 223 (32)  |           |           |           |           |
| Odd                   | 1318 (16) | 1672 (16) | 442 (30)  | 484 (30)  |           |           |           |           |
| offset                | 674*( 6)  | 677-( 6)  | 681-( 6)  | 681 ( 6)  | 683 ( 6)  | 683 ( 6)  | 841*(10)  | 890-(10)  |
|                       | 891 (10)  | 891 (10)  | 901-(10)  | 902 (10)  | 902 (10)  | 938*(10)  | 987-(10)  | 988 (10)  |
|                       | 988 (10)  | 998-(10)  | 999 (10)  | 999 (10)  | 1518*(10) | 1569-(10) | 1570 (10) | 1583-(10) |
|                       | 1584 (10) | 1618*(10) | 1669-(10) | 1670 (10) | 1683-(10) | 1684 (10) | 189*(12)  | 207-(12)  |
|                       | 241*(12)  | 271-(12)  | 281 (12)  | 284-(12)  | 284 (12)  | 316*(12)  | 320-(12)  | 326 (12)  |
|                       | 332-(12)  | 332 (12)  | 385*(12)  | 424*(12)  | 455-(12)  | 513*(12)  | 517-(12)  | 523 (12)  |
|                       | 529-(12)  | 529 (12)  | 2936*(13) | 2943*(13) | 2948*(13) | 2951*(13) | 75*(18)   | 78 (18)   |
|                       | 87*(18)   | 90 (18)   | 90 (18)   | 97*(18)   | 110 (18)  | 111-(18)  | 111 (18)  | 112 (18)  |
|                       | 122 (18)  | 129 (18)  | 137*(18)  | 139 (18)  | 671*(18)  | 674 (18)  | 1239*(18) | 1248 (18) |
|                       | 156*(23)  | 178*(23)  | 180 (23)  | 211 (23)  | 138*(28)  | 140-(28)  | 798*(28)  | 812-(28)  |
|                       | 815 (28)  | 817-(28)  | 817 (28)  | 937*(28)  | 939-(28)  | 47*(32)   | 85-(32)   | 912*(41)  |
|                       | 919-(41)  | 920 (41)  | 923 (41)  |           |           |           |           |           |
| OffsetPtr             | 335 ( 6)  | 1471 ( 6) | 1693 ( 6) | 1897 ( 6) | 2129 ( 6) | 2964 ( 6) | 3431 ( 6) | 229 (10)  |

|                   |            |            |            |            |            |            |            |            |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| OffsetPtrWStr     | 2948* (13) | 177 (15)   | 75* (18)   | 90 (18)    | 129 (18)   | 71 (20)    | 981 (21)   |            |
|                   | 809 (6)    | 891 (6)    | 1018 (6)   | 1168 (6)   | 2721 (6)   | 2259 (10)  | 2951* (13) | 87* (18)   |
|                   | 230 (18)   | 160 (19)   | 151 (21)   | 117 (40)   |            |            |            |            |
| OffsetRect        | 613 (19)   | 831 (19)   | 835 (19)   | 383 (21)   |            |            |            |            |
| OffsetRgn         | 3065 (16)  | 451 (18)   | 589 (18)   |            |            |            |            |            |
| OffsetVRect       | 896 (18)   | 1529 (18)  | 1630 (18)  | 1805 (18)  | 332 (20)   | 52* (42)   |            |            |
| oldBkColor        | 2249* (6)  | 2262 (6)   | 2276 (6)   | 2283 (6)   | 2296* (6)  | 2311 (6)   | 2323 (6)   |            |
| oldBRect          | 2879* (10) |            |            |            |            |            |            |            |
| oldCodeReserve    | 2188* (16) | 2202 (16)  | 2212 (16)  | 2213 (16)  | 2232 (16)  |            |            |            |
| oldColor          | 76* (6)    | 92= (6)    | 94 (6)     | 95= (6)    | 95 (6)     | 96 (6)     | 98 (6)     | 100 (6)    |
|                   | 114* (6)   | 128= (6)   | 129= (6)   | 130= (6)   | 131= (6)   | 132= (6)   | 133= (6)   | 134= (6)   |
|                   | 135= (6)   | 137 (6)    | 1255* (6)  | 1262 (6)   | 1292 (6)   | 2807* (6)  | 2816 (6)   | 2824 (6)   |
|                   | 297* (26)  | 313= (26)  | 315 (26)   | 316= (26)  | 316 (26)   | 317 (26)   | 319 (26)   | 321 (26)   |
|                   | 785* (26)  | 799= (26)  | 800= (26)  | 801= (26)  | 802= (26)  | 803= (26)  | 804= (26)  | 805= (26)  |
|                   | 806= (26)  | 808 (26)   |            |            |            |            |            |            |
| oldDevRes         | 500* (37)  | 517= (37)  | 566 (37)   |            |            |            |            |            |
| oldEffectiveRes   | 501* (37)  | 518= (37)  | 555 (37)   | 556 (37)   | 557 (37)   | 558 (37)   |            |            |
| oldElements       | 1488* (40) | 1499= (40) | 1508 (40)  |            |            |            |            |            |
| oldEnd            | 1079* (40) | 1135= (40) | 1139 (40)  |            |            |            |            |            |
| oldFillColor      | 2248* (6)  | 2262 (6)   | 2276 (6)   | 2283 (6)   | 2295* (6)  | 2311 (6)   | 2323 (6)   |            |
| oldFlag           | 1054* (17) |            |            |            |            |            |            |            |
| oldHText          | 80* (23)   | 96= (23)   | 104 (23)   |            |            |            |            |            |
| oldInterior       | 1626* (37) | 1634= (37) | 1648 (37)  | 1652 (37)  | 1661 (37)  |            |            |            |
| oldLongValue      | 1052* (21) | 1064= (21) | 1071 (21)  |            |            |            |            |            |
| oldMemReserve     | 2189* (16) | 2202 (16)  | 2212 (16)  | 2213 (16)  | 2232 (16)  |            |            |            |
| oldMessage        | 206* (7)   | 74* (8)    | 76 (8)     | 77= (8)    | 78 (8)     |            |            |            |
| oldMoreMast       | 337* (14)  | 362= (14)  | 363 (14)   | 365 (14)   |            |            |            |            |
| oldOffset         | 100* (18)  | 120= (18)  | 126 (18)   |            |            |            |            |            |
| oldPageAreas      | 498* (37)  | 515= (37)  | 564 (37)   | 565 (37)   |            |            |            |            |
| oldPerm           | 1231* (16) | 1241= (16) | 1241 (16)  | 1246= (16) | 1282= (16) | 1282 (16)  | 2032* (16) | 2085= (16) |
|                   | 2087= (16) | 2087 (16)  | 197* (28)  | 202= (28)  | 213 (28)   | 993* (28)  | 160* (33)  | 199= (33)  |
|                   | 205= (33)  | 205 (33)   |            |            |            |            |            |            |
| oldPort           | 1206* (40) | 1216 (40)  | 1229 (40)  |            |            |            |            |            |
| oldPrinterDev     | 499* (37)  | 516= (37)  | 567 (37)   |            |            |            |            |            |
| oldQDRect         | 2875* (10) |            |            |            |            |            |            |            |
| oldResLoad        | 103* (28)  | 110= (28)  | 128 (28)   | 152* (28)  | 161= (28)  | 184 (28)   | 410* (28)  | 447= (28)  |
|                   | 457 (28)   | 1116* (28) |            |            |            |            |            |            |
| oldRgn            | 274* (9)   | 488* (10)  | 507 (10)   | 512 (10)   | 532 (10)   |            |            |            |
| oldSelEnd         | 547* (40)  | 552= (40)  | 556 (40)   |            |            |            |            |            |
| oldSelStart       | 546* (40)  | 551= (40)  | 556 (40)   |            |            |            |            |            |
| oldSize           | 1517* (10) | 1559= (10) | 1573 (10)  | 1587 (10)  | 1617* (10) | 1659= (10) | 1673 (10)  | 1687 (10)  |
|                   | 423* (12)  | 450= (12)  | 461 (12)   | 1262* (40) | 1264= (40) | 1279 (40)  | 1279 (40)  | 708* (41)  |
|                   | 746= (41)  | 749 (41)   | 757 (41)   |            |            |            |            |            |
| oldStart          | 1078* (40) | 1134= (40) | 1139 (40)  |            |            |            |            |            |
| oldState          | 1533* (6)  | 1547= (6)  | 1550 (6)   | 1752* (6)  | 1767= (6)  | 1770 (6)   | 1952* (6)  | 1965= (6)  |
|                   | 1969 (6)   |            |            |            |            |            |            |            |
| oldStyles         | 1487* (40) | 1495= (40) | 1496 (40)  | 1498 (40)  | 1499 (40)  | 1507 (40)  |            |            |
| oldStyleSize      | 105* (39)  | 117= (39)  | 133= (39)  | 153 (39)   |            |            |            |            |
| oldTop            | 1249* (6)  | 1267= (6)  | 1270 (6)   |            |            |            |            |            |
| OldTrapAddr       | 20* (35)   | 30* (35)   | 159= (35)  | 159 (35)   | 194= (35)  | 194 (35)   |            |            |
| oldTrapAddr       | 765 (28)   | 65* (34)   | 84= (35)   | 244 (35)   |            |            |            |            |
| oldViewPerPage    | 1622* (37) | 1635= (37) | 1657 (37)  | 1662 (37)  |            |            |            |            |
| oldVRect          | 2877* (10) |            |            |            |            |            |            |            |
| oldVRefnum        | 630* (17)  | 673 (17)   | 680 (17)   | 584* (26)  | 637 (26)   | 642 (26)   |            |            |
| OneSubJob         | 324* (36)  | 1118* (37) | 1142= (37) | 1186 (37)  | 1506 (37)  |            |            |            |
| opcode            | 310* (30)  | 344= (30)  |            |            |            |            |            |            |
| Open              | 652 (6)    | 1544* (13) | 1949* (13) | 747 (16)   | 1452 (17)  | 1490* (18) | 1497 (18)  | 801* (19)  |
|                   | 808 (19)   |            |            |            |            |            |            |            |
| OpenAFile         | 1192* (13) | 702 (17)   | 720* (17)  | 725= (17)  | 848 (17)   | 1234 (17)  | 1334 (17)  |            |
| OpenAgain         | 1198* (13) | 2217 (16)  | 734* (17)  |            |            |            |            |            |
| OpenCPort         | 584 (14)   | 1843 (18)  |            |            |            |            |            |            |
| openData          | 1193* (13) | 721* (17)  | 725 (17)   | 414* (25)  | 579* (26)  | 623 (26)   |            |            |
| OpenDeskAcc       | 2104 (16)  | 2115 (16)  |            |            |            |            |            |            |
| OpenDeskAccessory | 763* (13)  | 1999 (16)  | 2024* (16) |            |            |            |            |            |
| openDocWindows    | 399* (19)  | 408= (19)  | 408 (19)   | 417= (19)  | 420 (19)   |            |            |            |
| OpenFile          | 725 (17)   | 413* (25)  | 579* (26)  | 595= (26)  | 596 (26)   | 666= (26)  |            |            |
| openingDoc        | 1199* (13) | 734* (17)  |            |            |            |            |            |            |
| openInitially     | 1875* (13) | 1898* (13) | 137 (19)   | 195= (19)  |            |            |            |            |
| OpenNew           | 852* (13)  | 731 (16)   | 1482 (16)  | 2130* (16) |            |            |            |            |
| OpenOld           | 859* (13)  | 736 (16)   | 1508 (16)  | 2183* (16) |            |            |            |            |
| OpenPicture       | 1848 (18)  |            |            |            |            |            |            |            |
| OpenPort          | 586 (14)   | 1845 (18)  |            |            |            |            |            |            |
| OpenPrintShop     | 270* (36)  | 641 (37)   | 1203* (37) |            |            |            |            |            |
| OpenResFile       | 640 (26)   |            |            |            |            |            |            |            |
| OpenRFPPerm       | 653 (26)   |            |            |            |            |            |            |            |
| OpenRgn           | 446 (18)   |            |            |            |            |            |            |            |
| openRsrc          | 1193* (13) | 721* (17)  | 725 (17)   | 414* (25)  | 579* (26)  | 632 (26)   |            |            |
| OpenSubView       | 1495* (18) | 1501 (18)  |            |            |            |            |            |            |
| opString          | 172* (14)  | 183= (14)  | 189 (14)   | 198 (14)   | 202 (14)   | 217 (14)   | 219 (14)   |            |
| ORD               | 136 (4)    | 889 (6)    | 958 (6)    | 967 (6)    | 977 (6)    | 1017 (6)   | 1087 (6)   | 1096 (6)   |
|                   | 1106 (6)   | 123 (12)   | 440 (12)   | 154 (15)   | 155 (15)   | 1683 (16)  | 609 (21)   | 614 (21)   |
|                   | 649 (21)   | 663 (21)   | 764 (21)   | 409 (26)   | 419 (26)   | 419 (26)   | 426 (26)   | 432 (26)   |
|                   | 436 (26)   | 440 (26)   | 940 (26)   | 1116 (26)  | 1135 (26)  | 578 (28)   | 580 (28)   | 89 (33)    |
|                   | 91 (33)    | 103 (33)   | 205 (32)   | 307 (32)   | 422 (37)   | 1140 (37)  | 331 (40)   | 499 (40)   |
|                   | 59 (41)    | 679 (41)   | 733 (41)   | 756 (41)   |            |            |            |            |
| Ord               | 891 (10)   | 902 (10)   | 988 (10)   | 999 (10)   | 157 (12)   | 207 (12)   | 281 (12)   | 326 (12)   |
|                   | 523 (12)   | 542 (12)   | 559 (12)   | 2046 (16)  | 166 (28)   | 174 (28)   | 970 (28)   | 1048 (28)  |
|                   | 55 (32)    | 57 (32)    | 57 (32)    | 68 (32)    | 73 (32)    | 78 (32)    | 81 (32)    | 88 (32)    |
|                   | 91 (32)    |            |            |            |            |            |            |            |
| ORD4              | 682 (6)    | 682 (6)    | 1287 (6)   | 2350 (6)   | 2819 (6)   | 3167 (6)   | 1111 (18)  | 121 (19)   |
|                   | 124 (19)   | 652 (19)   | 130 (24)   | 435 (26)   | 971 (26)   | 979 (26)   | 997 (26)   | 1322 (26)  |
|                   | 815 (28)   | 221 (30)   | 224 (30)   | 223 (32)   | 226 (32)   | 121 (35)   | 131 (35)   | 165 (35)   |
|                   | 200 (35)   | 486 (37)   | 381 (40)   | 383 (40)   | 588 (40)   | 629 (40)   |            |            |
| Ord4              | 209 (4)    | 749 (14)   |            |            |            |            |            |            |
| ordObject         | 24* (12)   | 147* (31)  | 11* (33)   | 12* (33)   | 135* (33)  | 137 (33)   | 138 (33)   | 139 (33)   |
| origCursor        | 13* (4)    | 215 (4)    | 298= (4)   |            |            |            |            |            |
| origCursor        | 11* (4)    | 124= (4)   | 217 (4)    | 325= (4)   |            |            |            |            |
| origView          | 108* (5)   | 186* (5)   | 226* (5)   | 273* (5)   | 447* (6)   | 453 (6)    | 457 (6)    | 460 (6)    |
|                   | 460 (6)    | 462 (6)    | 463 (6)    | 466 (6)    | 936* (6)   | 940 (6)    | 1063* (6)  | 1068 (6)   |

|           |           |           |           |           |           |           |           |           |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| ortho     | 1222*( 6) | 1230 ( 6) | 1236 ( 6) | 1238 ( 6) | 1556*(13) | 746*(18)  | 749 (18)  |           |
|           | 153*(20)  | 163-(20)  | 167 (20)  | 167 (20)  | 170 (20)  | 439*(37)  | 458-(37)  | 459 (37)  |
|           | 463 (37)  | 463 (37)  | 463 (37)  | 463 (37)  | 1228*(37) | 1233-(37) | 1235 (37) | 1235 (37) |
| orthoVHS  | 1237 (37) | 1237 (37) | 2035*(37) | 2038-(37) | 2039 (37) | 2040 (37) |           |           |
|           | 412*(37)  | 415-(37)  | 417 (37)  | 417 (37)  | 423 (37)  | 976*(37)  | 978-(37)  | 979 (37)  |
| orthoVhs  | 980 (37)  | 980 (37)  | 981 (37)  |           |           |           |           |           |
| orthovhs  | 576*(40)  | 585-(40)  | 588 (40)  | 594 (40)  | 605 (40)  | 629 (40)  | 630 (40)  | 631 (40)  |
| OSErr     | 715*(37)  | 728 (37)  | 734 (37)  | 734 (37)  | 757-(37)  |           |           |           |
|           | 21 ( 2)   | 199 ( 2)  | 172 ( 4)  | 265 ( 6)  | 308 ( 6)  | 1129 ( 6) | 1250 ( 6) | 1399 ( 6) |
|           | 1441 ( 6) | 1622 ( 6) | 1664 ( 6) | 1843 ( 6) | 1878 ( 6) | 2035 ( 6) | 2089 ( 6) | 2493 ( 6) |
|           | 2678 ( 6) | 2782 ( 6) | 36 ( 8)   | 115 ( 8)  | 259 (12)  | 1194 (13) | 1391 (13) | 2895 (13) |
|           | 2905 (13) | 155 (14)  | 174 (14)  | 300 (14)  | 1117 (14) | 117 (15)  | 279 (16)  | 286 (16)  |
|           | 377 (16)  | 713 (16)  | 1389 (16) | 1445 (16) | 1465 (16) | 1659 (16) | 1962 (16) | 2035 (16) |
|           | 2037 (16) | 2495 (16) | 2946 (16) | 19 (17)   | 157 (17)  | 242 (17)  | 244 (17)  | 302 (17)  |
|           | 308 (17)  | 474 (17)  | 519 (17)  | 627 (17)  | 636 (17)  | 637 (17)  | 722 (17)  | 914 (17)  |
|           | 964 (17)  | 1056 (17) | 1253 (17) | 1295 (17) | 1300 (17) | 1301 (17) | 1314 (17) | 1324 (17) |
|           | 185 (18)  | 431 (18)  | 27 (19)   | 96 (19)   | 214 (20)  | 631 (21)  | 1213 (21) | 1249 (21) |
|           | 165 (23)  | 380 (25)  | 386 (25)  | 393 (25)  | 402 (25)  | 406 (25)  | 415 (25)  | 25 (26)   |
|           | 28 (26)   | 95 (26)   | 133 (26)  | 135 (26)  | 194 (26)  | 204 (26)  | 233 (26)  | 234 (26)  |
|           | 581 (26)  | 585 (26)  | 590 (26)  | 903 (28)  | 161 (33)  | 99 (35)   | 144 (35)  | 178 (35)  |
|           | 111 (37)  | 175 (37)  | 228 (37)  | 253 (37)  | 607 (37)  | 631 (37)  | 1126 (37) | 1131 (37) |
|           | 1206 (37) | 1282 (37) | 1438 (37) | 1458 (37) | 1531 (37) | 1587 (37) | 1710 (37) | 2073 (37) |
|           | 2099 (37) | 1082 (40) | 1089 (40) | 1209 (40) | 22 (41)   | 357 (41)  | 447 (41)  | 639 (41)  |
|           | 824 (41)  |           |           |           |           |           |           |           |
| OSerr     | 1065 (13) |           |           |           |           |           |           |           |
| OSIntf    | 14*( 1)   | 58*( 3)   | 14*( 5)   | 136*( 7)  | 25*( 9)   | 17*(11)   | 45*(13)   | 44*(25)   |
|           | 130*(27)  | 110*(29)  | 15*(31)   | 53*(34)   | 14*(36)   | 21*(38)   | 14*(42)   |           |
| OSTrap    | 459 (26)  |           |           |           |           |           |           |           |
| OSType    | 553 (13)  | 687 (13)  | 1104 (13) | 1105 (13) | 1140 (13) | 2710 (13) | 58 (16)   | 10 (17)   |
|           | 890 (26)  |           |           |           |           |           |           |           |
| otherDoc  | 2187*(16) | 2215-(16) | 2216 (16) | 2217 (16) | 913*(17)  | 938-(17)  | 939 (17)  | 940 (17)  |
|           | 1058*(17) |           |           |           |           |           |           |           |
| ourHeap   | 2033*(16) | 2099-(16) | 2103 (16) |           |           |           |           |           |
| outer     | 320*(25)  | 727*(26)  | 730 (26)  | 730 (26)  | 731 (26)  | 731 (26)  |           |           |
| outerRect | 479*(14)  | 524-(14)  | 525 (14)  | 526 (14)  | 526 (14)  | 527 (14)  |           |           |
| outline   | 1037 (26) |           |           |           |           |           |           |           |
| OVERRIDE  | 35*( 1)   | 40*( 1)   | 51*( 1)   | 65*( 1)   | 88*( 1)   | 43*( 2)   | 57*( 2)   | 109*( 2)  |
|           | 357*( 2)  | 73*( 5)   | 77*( 5)   | 79*( 5)   | 82*( 5)   | 88*( 5)   | 92*( 5)   | 96*( 5)   |
|           | 102*( 5)  | 108*( 5)  | 124*( 5)  | 147*( 5)  | 151*( 5)  | 153*( 5)  | 177*( 5)  | 181*( 5)  |
|           | 183*( 5)  | 186*( 5)  | 217*( 5)  | 221*( 5)  | 223*( 5)  | 226*( 5)  | 260*( 5)  | 264*( 5)  |
|           | 266*( 5)  | 269*( 5)  | 271*( 5)  | 273*( 5)  | 285*( 5)  | 312*( 5)  | 316*( 5)  | 318*( 5)  |
|           | 321*( 5)  | 323*( 5)  | 331*( 5)  | 358*( 5)  | 362*( 5)  | 364*( 5)  | 367*( 5)  | 369*( 5)  |
|           | 378*( 5)  | 403*( 5)  | 407*( 5)  | 409*( 5)  | 412*( 5)  | 414*( 5)  | 422*( 5)  | 452*( 5)  |
|           | 456*( 5)  | 458*( 5)  | 461*( 5)  | 470*( 5)  | 472*( 5)  | 491*( 5)  | 519*( 5)  | 523*( 5)  |
|           | 525*( 5)  | 528*( 5)  | 532*( 5)  | 547*( 5)  | 573*( 5)  | 577*( 5)  | 579*( 5)  | 582*( 5)  |
|           | 584*( 5)  | 586*( 5)  | 589*( 5)  | 591*( 5)  | 593*( 5)  | 595*( 5)  | 597*( 5)  | 600*( 5)  |
|           | 602*( 5)  | 604*( 5)  | 606*( 5)  | 610*( 5)  | 614*( 5)  | 620*( 5)  | 624*( 5)  | 649*( 5)  |
|           | 653*( 5)  | 655*( 5)  | 662*( 5)  | 666*( 5)  | 677*( 5)  | 681*( 5)  | 683*( 5)  | 685*( 5)  |
|           | 689*( 5)  | 298*( 6)  | 345*( 6)  | 367*( 6)  | 379*( 6)  | 420*( 6)  | 447*( 6)  | 474*( 6)  |
|           | 494*( 6)  | 561*( 6)  | 756*( 6)  | 796*( 6)  | 819*( 6)  | 844*( 6)  | 878*( 6)  | 901*( 6)  |
|           | 924*( 6)  | 1006*( 6) | 1028*( 6) | 1051*( 6) | 1158*( 6) | 1178*( 6) | 1200*( 6) | 1212*( 6) |
|           | 1222*( 6) | 1245*( 6) | 1369*( 6) | 1433*( 6) | 1481*( 6) | 1509*( 6) | 1521*( 6) | 1531*( 6) |
|           | 1592*( 6) | 1657*( 6) | 1703*( 6) | 1728*( 6) | 1740*( 6) | 1750*( 6) | 1812*( 6) | 1870*( 6) |
|           | 1907*( 6) | 1928*( 6) | 1940*( 6) | 1950*( 6) | 2004*( 6) | 2080*( 6) | 2139*( 6) | 2165*( 6) |
|           | 2177*( 6) | 2240*( 6) | 2291*( 6) | 2529*( 6) | 2550*( 6) | 2563*( 6) | 2616*( 6) | 2636*( 6) |
|           | 2648*( 6) | 2708*( 6) | 2731*( 6) | 2757*( 6) | 2769*( 6) | 2803*( 6) | 2904*( 6) | 2951*( 6) |
|           | 2974*( 6) | 2995*( 6) | 3007*( 6) | 3018*( 6) | 3039*( 6) | 3062*( 6) | 3088*( 6) | 3102*( 6) |
|           | 3113*( 6) | 3136*( 6) | 3151*( 6) | 3175*( 6) | 3218*( 6) | 3231*( 6) | 3281*( 6) | 3312*( 6) |
|           | 3415*( 6) | 3441*( 6) | 3464*( 6) | 3509*( 6) | 3542*( 6) | 194*( 9)  | 199*( 9)  | 202*( 9)  |
|           | 206*( 9)  | 217*( 9)  | 220*( 9)  | 224*( 9)  | 227*( 9)  | 400*( 9)  | 452*( 9)  | 457*( 9)  |
|           | 460*( 9)  | 479*( 9)  | 482*( 9)  | 497*( 9)  | 543*( 9)  | 565*( 9)  | 588*( 9)  | 600*( 9)  |
|           | 603*( 9)  | 625*( 9)  | 650*( 9)  | 654*( 9)  | 662*( 9)  | 687*( 9)  | 692*( 9)  | 700*( 9)  |
|           | 723*( 9)  | 746*( 9)  | 769*( 9)  | 140*(10)  | 240*(10)  | 277*(10)  | 290*(10)  | 351*(10)  |
|           | 542*(10)  | 620*(10)  | 686*(10)  | 2140*(10) | 2234*(10) | 2270*(10) | 2302*(10) | 2315*(10) |
|           | 2334*(10) | 2406*(10) | 2472*(10) | 2496*(10) | 2627*(10) | 2693*(10) | 2710*(10) | 2762*(10) |
|           | 2821*(10) | 2833*(10) | 2931*(10) | 3030*(10) | 3059*(10) | 3094*(10) | 3127*(10) | 3162*(10) |
|           | 246*(11)  | 249*(11)  | 260*(11)  | 622*(12)  | 651*(12)  | 677*(12)  | 594*(13)  | 609*(13)  |
|           | 695*(13)  | 699*(13)  | 943*(13)  | 995*(13)  | 997*(13)  | 1061*(13) | 1143*(13) | 1157*(13) |
|           | 1161*(13) | 1190*(13) | 1404*(13) | 1405*(13) | 1488*(13) | 1702*(13) | 1717*(13) | 1733*(13) |
|           | 1735*(13) | 1776*(13) | 1780*(13) | 1782*(13) | 1785*(13) | 1789*(13) | 1791*(13) | 1795*(13) |
|           | 1799*(13) | 1801*(13) | 1805*(13) | 1809*(13) | 1813*(13) | 1817*(13) | 1846*(13) | 1850*(13) |
|           | 1857*(13) | 1936*(13) | 1940*(13) | 1942*(13) | 1945*(13) | 1949*(13) | 1951*(13) | 1957*(13) |
|           | 1961*(13) | 1967*(13) | 1973*(13) | 1975*(13) | 1979*(13) | 1981*(13) | 1990*(13) | 1996*(13) |
|           | 2000*(13) | 2002*(13) | 2016*(13) | 2020*(13) | 2033*(13) | 2035*(13) | 2078*(13) | 2082*(13) |
|           | 2084*(13) | 2087*(13) | 2089*(13) | 2091*(13) | 2100*(13) | 2102*(13) | 2104*(13) | 2120*(13) |
|           | 2140*(13) | 2159*(13) | 2161*(13) | 2163*(13) | 2166*(13) | 2168*(13) | 2173*(13) | 2176*(13) |
|           | 2178*(13) | 2188*(13) | 2190*(13) | 2207*(13) | 2237*(13) | 2241*(13) | 2243*(13) | 2249*(13) |
|           | 2272*(13) | 2291*(13) | 2295*(13) | 2297*(13) | 2300*(13) | 2302*(13) | 2305*(13) | 2307*(13) |
|           | 2309*(13) | 2316*(13) | 2344*(13) | 2348*(13) | 2468*(13) | 2536*(13) | 2539*(13) | 2542*(13) |
|           | 2565*(13) | 2568*(13) | 2571*(13) | 2572*(13) | 2573*(13) | 2574*(13) | 2578*(13) | 2582*(13) |
|           | 23*(15)   | 670*(16)  | 908*(16)  | 101*(17)  | 503*(17)  | 605*(17)  | 1103*(18) | 1904*(18) |
|           | 171*(19)  | 214*(19)  | 226*(19)  | 376*(19)  | 432*(19)  | 444*(19)  | 455*(19)  | 502*(19)  |
|           | 535*(19)  | 649*(19)  | 662*(19)  | 673*(19)  | 687*(19)  | 747*(19)  | 761*(19)  | 801*(19)  |
|           | 816*(19)  | 925*(19)  | 1025*(19) | 80*(20)   | 111*(20)  | 122*(20)  | 137*(20)  | 296*(20)  |
|           | 328*(20)  | 364*(20)  | 374*(20)  | 386*(20)  | 397*(20)  | 429*(20)  | 651*(20)  | 661*(20)  |
|           | 55*(21)   | 66*(21)   | 77*(21)   | 162*(21)  | 204*(21)  | 216*(21)  | 226*(21)  | 248*(21)  |
|           | 303*(21)  | 317*(21)  | 337*(21)  | 424*(21)  | 504*(21)  | 567*(21)  | 579*(21)  | 590*(21)  |
|           | 602*(21)  | 659*(21)  | 677*(21)  | 690*(21)  | 760*(21)  | 777*(21)  | 924*(21)  | 962*(21)  |
|           | 991*(21)  | 1017*(21) | 1049*(21) | 1182*(21) | 1244*(21) | 1273*(21) | 1285*(21) | 1296*(21) |
|           | 1319*(21) | 1342*(21) | 1384*(21) | 1432*(21) | 121*(22)  | 42*(23)   | 63*(23)   | 74*(23)   |
|           | 282*(23)  | 297*(23)  | 309*(23)  | 120*(24)  | 201*(36)  | 204*(36)  | 211*(36)  | 214*(36)  |
|           | 220*(36)  | 279*(36)  | 315*(36)  | 320*(36)  | 339*(36)  | 348*(36)  | 353*(36)  | 361*(36)  |
|           | 373*(36)  | 377*(36)  | 380*(36)  | 401*(36)  | 404*(36)  | 410*(36)  | 424*(36)  | 428*(36)  |
|           | 462*(36)  | 468*(36)  | 493*(36)  | 498*(36)  | 499*(36)  | 500*(36)  | 505*(36)  | 316*(37)  |
|           | 408*(37)  | 436*(37)  | 479*(37)  | 496*(37)  | 654*(37)  | 693*(37)  | 712*(37)  | 770*(37)  |
|           | 929*(37)  | 1092*(37) | 1107*(37) | 1427*(37) | 1559*(37) | 1617*(37) | 1881*(37) | 1909*(37) |
|           | 1923*(37) | 2048*(37) | 2124*(37) | 2138*(37) | 2165*(37) | 2177*(37) | 160*(38)  | 165*(38)  |
|           | 167*(38)  | 170*(38)  | 179*(38)  | 192*(38)  | 196*(38)  | 202*(38)  | 205*(38)  | 211*(38)  |
|           | 215*(38)  | 218*(38)  | 221*(38)  | 227*(38)  | 230*(38)  | 233*(38)  | 240*(38)  | 243*(38)  |

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                     | 256*(38)  | 259*(38)  | 262*(38)  | 336*(38)  | 340*(38)  | 343*(38)  | 347*(38)  | 353*(38)  |
|                     | 358*(38)  | 403*(38)  | 408*(38)  | 411*(38)  | 415*(38)  | 450*(38)  | 469*(38)  | 474*(38)  |
|                     | 478*(38)  | 523*(38)  | 526*(38)  | 528*(38)  | 534*(38)  | 557*(38)  | 574*(38)  | 577*(38)  |
|                     | 588*(38)  | 70*(40)   | 127*(40)  | 164*(40)  | 176*(40)  | 214*(40)  | 240*(40)  | 265*(40)  |
|                     | 366*(40)  | 563*(40)  | 574*(40)  | 641*(40)  | 675*(40)  | 695*(40)  | 828*(40)  | 884*(40)  |
|                     | 912*(40)  | 934*(40)  | 950*(40)  | 982*(40)  | 1059*(40) | 1076*(40) | 1178*(40) | 1258*(40) |
|                     | 1452*(40) | 1598*(40) | 1620*(40) | 1635*(40) | 93*(41)   | 163*(41)  | 263*(41)  | 279*(41)  |
|                     | 299*(41)  | 333*(41)  | 345*(41)  | 416*(41)  | 567*(41)  | 582*(41)  | 599*(41)  | 612*(41)  |
|                     | 665*(41)  | 880*(41)  | 894*(41)  | 939*(41)  |           |           |           |           |
| -P-                 |           |           |           |           |           |           |           |           |
| P                   | 214*( 7)  | 26*( 8)   | 62*( 8)   | 65 ( 8)   | 80*(12)   | 106*(12)  | 132-(12)  | 133-(12)  |
|                     | 153*(12)  | 157-(12)  | 160 (12)  | 164-(12)  | 164 (12)  | 168-(12)  | 168 (12)  | 169 (12)  |
|                     | 174 (12)  | 187*(12)  | 192-(12)  | 196 (12)  | 196 (12)  | 197-(12)  | 197 (12)  | 199 (12)  |
|                     | 204-(12)  | 207 (12)  | 382*(12)  | 388-(12)  | 393 (12)  | 393 (12)  | 394-(12)  | 394 (12)  |
|                     | 395 (12)  | 681*(12)  | 730*(12)  | 2927*(13) | 2935*(13) | 2942*(13) | 2948*(13) | 2951*(13) |
|                     | 41*(18)   | 55-(18)   | 75*(18)   | 78-(18)   | 78 (18)   | 79 (18)   | 80-(18)   | 80 (18)   |
|                     | 87*(18)   | 90 (18)   | 90 (18)   | 97*(18)   | 109 (18)  | 126-(18)  | 128 (18)  | 129 (18)  |
|                     | 136*(18)  | 139 (18)  | 923*(26)  | 927 (26)  | 1149*(26) | 1157-(26) | 1159-(26) | 1161-(26) |
|                     | 151*(28)  | 164-(28)  | 165 (28)  | 166-(28)  | 166 (28)  | 172 (28)  | 172 (28)  | 174-(28)  |
|                     | 174 (28)  | 174 (28)  | 801*(28)  | 815-(28)  | 816 (28)  | 816 (28)  | 962*(28)  | 966-(28)  |
|                     | 969 (28)  | 970-(28)  | 970 (28)  | 62*(30)   | 70-(30)   | 73 (30)   | 103*(30)  | 119-(30)  |
|                     | 123 (30)  | 66*(33)   | 99-(33)   | 102-(33)  | 103-(33)  | 103 (33)  | 44*(32)   | 54-(32)   |
|                     | 55 (32)   | 57 (32)   | 59 (32)   | 62 (32)   | 88-(32)   | 88 (32)   | 91-(32)   | 91 (32)   |
| PackIntf            | 14*( 1)   | 58*( 3)   | 14*( 5)   | 136*( 7)  | 25*( 9)   | 17*(11)   | 45*(13)   | 44*(25)   |
|                     | 130*(27)  | 110*(29)  | 15*(31)   | 53*(34)   | 14*(36)   | 21*(38)   | 14*(42)   |           |
| PageAreas           | 81*(36)   | 94 (36)   | 498 (37)  |           |           |           |           |           |
| PageInteriorChanged | 1724*(13) | 1508*(18) | 1653 (37) |           |           |           |           |           |
| pageNumber          | 2375*(13) | 2415*(13) | 141*(24)  | 211*(24)  | 249*(36)  | 319*(36)  | 348*(36)  | 1092*(37) |
|                     | 1224*(37) | 1231 (37) | 1881*(37) | 1895 (37) |           |           |           |           |
| pageNumStyle        | 719*(37)  | 747 (37)  | 748 (37)  |           |           |           |           |           |
| pagesPerSubjob      | 301*(36)  | 589*(37)  | 596-(37)  | 1445*(37) | 1482 (37) | 1496-(37) | 1500-(37) | 1500 (37) |
|                     | 1507 (37) | 1508 (37) | 1514-(37) | 1518 (37) | 1520 (37) |           |           |           |
| pageStrip           | 1254*(37) | 1258 (37) | 1261 (37) | 1266 (37) | 1269 (37) |           |           |           |
| pageStrips          | 1684*(13) | 2364*(13) | 717*(18)  | 719 (18)  | 56*(24)   | 211*(36)  | 436*(37)  | 462 (37)  |
|                     | 469 (37)  | 1452*(37) | 1468 (37) | 1469 (37) | 1624*(37) | 1665 (37) | 1666 (37) |           |
| PageToStrip         | 249*(36)  | 1224*(37) | 1239-(37) | 1852 (37) |           |           |           |           |
| PaintRect           | 281 (21)  |           |           |           |           |           |           |           |
| PaintRgn            | 453 (18)  |           |           |           |           |           |           |           |
| PAppFile            | 548*(13)  | 843 (13)  | 1883 (16) |           |           |           |           |           |
| paramBlock          | 319*(16)  | 339*(16)  | 361 (16)  | 362 (16)  |           |           |           |           |
| paramErr            | 615 (26)  | 616 (26)  | 650 (26)  | 651 (26)  |           |           |           |           |
| ParamText           | 217 (14)  | 167 (17)  | 332 (17)  | 739 (17)  | 771 (17)  | 1157 (17) |           |           |
| ParamTxt            | 112*( 5)  | 633*( 6)  |           |           |           |           |           |           |
| parentView          | 650*(13)  | 105*(15)  | 124 (15)  | 163 (15)  | 164 (15)  | 166 (15)  | 184 (15)  | 185-(15)  |
| parmDirID           | 276*(16)  | 297 (16)  | 308 (16)  |           |           |           |           |           |
| parmVRefnum         | 277*(16)  | 296 (16)  | 307-(16)  | 308 (16)  |           |           |           |           |
| ParseTitleTemplate  | 2875*(13) | 1063*(14) | 1110-(14) | 2819 (16) | 1469 (17) | 66 (19)   | 142 (19)  | 1402 (37) |
| partCode            | 1835*(13) | 1969*(13) | 2265*(13) | 2309*(13) | 1002*(19) | 1011 (19) | 551*(20)  | 561 (20)  |
|                     | 576 (20)  | 582 (20)  | 582 (20)  | 12*(21)   | 18 (21)   | 21 (21)   | 21 (21)   | 25 (21)   |
|                     | 1051*(21) | 1059-(21) | 1169*(21) | 1344*(21) | 1404*(21) | 1415 (21) |           |           |
| partialJob          | 326*(36)  | 1119*(37) |           |           |           |           |           |           |
| pass                | 1123*(37) | 1165-(37) |           |           |           |           |           |           |
| patBic              | 280 (21)  |           |           |           |           |           |           |           |
| patchBlock          | 101*(35)  | 109-(35)  | 111 (35)  | 114 (35)  | 121 (35)  | 130-(35)  | 147*(35)  | 151-(35)  |
|                     | 153 (35)  | 156 (35)  | 165 (35)  | 181*(35)  | 185-(35)  | 187 (35)  | 190 (35)  | 200 (35)  |
| PatchTrap           | 516 (28)  | 77*(34)   | 98*(35)   | 124-(35)  | 132-(35)  |           |           |           |
| patCopy             | 188 (37)  |           |           |           |           |           |           |           |
| PatHandle           | 1769 ( 6) |           |           |           |           |           |           |           |
| Pattern             | 1206 (14) | 891 (26)  |           |           |           |           |           |           |
| pattern             | 1073*(14) | 1079 (14) | 1082 (14) |           |           |           |           |           |
| PatternTemplate     | 339 ( 5)  | 340*( 5)  | 1693 ( 6) | 1711 ( 6) |           |           |           |           |
| PatternTemplatePtr  | 339*( 5)  | 1673 ( 6) | 1706 ( 6) | 1711 ( 6) |           |           |           |           |
| PatXOR              | 3194 (16) |           |           |           |           |           |           |           |
| patXOR              | 1209 (14) |           |           |           |           |           |           |           |
| patXor              | 590 (10)  | 595 (10)  | 603 (10)  |           |           |           |           |           |
| pb                  | 243*(17)  | 248 (17)  | 250 (17)  | 252 (17)  | 393*(25)  | 173*(26)  | 176 (26)  | 177 (26)  |
|                     | 194*(26)  | 197 (26)  | 197 (26)  | 205*(26)  | 210 (26)  | 218 (26)  | 219 (26)  | 220 (26)  |
|                     | 583*(26)  | 602 (26)  | 610 (26)  | 613 (26)  | 618 (26)  | 619 (26)  | 624 (26)  | 628 (26)  |
|                     | 648 (26)  | 657 (26)  |           |           |           |           |           |           |
| PBGetCatInfo        | 536 (17)  |           |           |           |           |           |           |           |
| PBGetFInfo          | 539 (17)  |           |           |           |           |           |           |           |
| PBGetWDInfo         | 218 (26)  |           |           |           |           |           |           |           |
| PBHDelete           | 147 (26)  |           |           |           |           |           |           |           |
| PBHGetFInfo         | 245 (26)  |           |           |           |           |           |           |           |
| PBHGetVInfo         | 1117 (17) |           |           |           |           |           |           |           |
| PBHOOpen            | 619 (26)  |           |           |           |           |           |           |           |
| PBHOOpenDeny        | 613 (26)  |           |           |           |           |           |           |           |
| PBHOOpenRFDeny      | 648 (26)  |           |           |           |           |           |           |           |
| PBoolArray          | 15 (28)   | 16*(28)   |           |           |           |           |           |           |
| PBoolean            | 136*(25)  |           |           |           |           |           |           |           |
| PBSetCatInfo        | 689 (17)  |           |           |           |           |           |           |           |
| PBSetFInfo          | 697 (17)  |           |           |           |           |           |           |           |
| PByte               | 137*(25)  | 279 (26)  | 392 (26)  | 409 (26)  | 436 (26)  | 440 (26)  | 769 (26)  | 151 (28)  |
|                     | 164 (28)  | 166 (28)  | 174 (28)  | 801 (28)  | 815 (28)  | 69 (32)   |           |           |
| pc                  | 145*( 8)  | 158-( 8)  | 159 ( 8)  | 393*(26)  | 417-(26)  | 419 (26)  | 419 (26)  | 425 (26)  |
|                     | 426-(26)  | 426 (26)  | 429 (26)  | 430 (26)  | 431 (26)  | 432-(26)  | 432 (26)  | 434 (26)  |
|                     | 435-(26)  | 435 (26)  | 436 (26)  | 440 (26)  |           |           |           |           |
| PCmdArray           | 18*(30)   | 19 (30)   | 62 (30)   | 103 (30)  |           |           |           |           |
| pcPtr               | 1910*( 6) | 1915-( 6) | 1921 ( 6) |           |           |           |           |           |
| peekEvent           | 3109*(16) | 3336 (16) | 3338 (16) |           |           |           |           |           |
| pegPoint            | 1884*(37) | 1895 (37) | 1898 (37) |           |           |           |           |           |
| PenMode             | 590 (10)  | 595 (10)  | 603 (10)  | 1224 (14) | 3194 (16) | 280 (21)  |           |           |
| PenNormal           | 1260 ( 6) | 1539 ( 6) | 1758 ( 6) | 1967 ( 6) | 2398 ( 6) | 2815 ( 6) | 580 (10)  | 2379 (10) |
|                     | 3193 (16) | 480 (18)  | 488 (19)  | 702 (21)  | 1392 (21) |           |           |           |
| PenPat              | 1225 (14) | 279 (21)  |           |           |           |           |           |           |
| PenSize             | 1261 ( 6) | 481 (18)  | 377 (37)  | 381 (37)  |           |           |           |           |
| PenState            | 399 (18)  | 514 (36)  |           |           |           |           |           |           |
| pEntry              | 730*(14)  | 749-(14)  | 751 (14)  |           |           |           |           |           |

|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| PerformCommand      | 971*(13)  | 1418 (16) | 2244*(16) |           |           |           |           |           |  |
| PerformMenuSetup    | 2919 (16) | 270*(29)  | 417*(30)  |           |           |           |           |           |  |
| perm                | 2394*(16) | 2402*(16) | 2404 (16) |           |           |           |           |           |  |
| PermAllocation      | 452 (12)  | 459 (12)  | 1241 (16) | 1246 (16) | 1282 (16) | 2085 (16) | 2087 (16) | 2402 (16) |  |
|                     | 1247 (18) | 1249 (18) | 210 (23)  | 212 (23)  | 331*(27)  | 855 (28)  | 857 (28)  | 867*(28)  |  |
|                     | 872*(28)  | 199 (33)  | 205 (33)  | 1116 (40) | 1119 (40) | 1146 (40) | 1153 (40) | 449 (41)  |  |
|                     | 500 (41)  | 508 (41)  | 826 (41)  | 837 (41)  | 852 (41)  |           |           |           |  |
| permanent           | 331*(27)  | 867*(28)  | 874 (28)  | 876 (28)  | 878 (28)  |           |           |           |  |
| permFlag            | 425*(12)  | 452*(12)  | 459*(12)  | 459 (12)  |           |           |           |           |  |
| PErrRecord          | 724*(14)  | 730 (14)  | 749 (14)  |           |           |           |           |           |  |
| PEventRecord        | 498*(13)  | 501 (13)  | 615 (13)  | 976 (13)  | 213 (15)  | 577 (16)  |           |           |  |
| pf                  | 21*( 8)   | 143*( 8)  | 153*( 8)  | 155 ( 8)  | 158 ( 8)  | 171 ( 8)  | 172 ( 8)  | 173 ( 8)  |  |
|                     | 174 ( 8)  |           |           |           |           |           |           |           |  |
| PFailInfo           | 150*( 7)  | 157 ( 7)  | 10 ( 8)   | 21 ( 8)   | 143 ( 8)  |           |           |           |  |
| pFileName           | 1490*(37) |           |           |           |           |           |           |           |  |
| pgStr               | 352*(37)  | 361 (37)  | 365 (37)  |           |           |           |           |           |  |
| phAboutApp          | 315*(13)  | 752 (16)  |           |           |           |           |           |           |  |
| PHandleList         | 152 (27)  | 153*(27)  |           |           |           |           |           |           |  |
| phase               | 582*( 5)  | 3007*( 6) | 3011 ( 6) | 613*(13)  | 933*(13)  | 224*(15)  | 1745*(16) | 1763 (16) |  |
|                     | 1765 (16) | 1776 (16) | 1799 (16) | 211*(38)  | 675*(40)  |           |           |           |  |
| phCmdErr            | 295*(13)  | 188 (14)  |           |           |           |           |           |           |  |
| phFileChanged       | 301*(13)  | 168 (17)  |           |           |           |           |           |           |  |
| phFinderPrintDialog | 57*(36)   | 1384 (37) |           |           |           |           |           |           |  |
| phGenError          | 294 (13)  | 181 (14)  |           |           |           |           |           |           |  |
| pHndl               | 1833*(18) | 1848*(18) | 1850 (18) | 1863 (18) | 1868 (18) | 1872 (18) | 1873 (18) |           |  |
| phNoPages           | 62*(36)   | 1475 (37) |           |           |           |           |           |           |  |
| phOfferReadOnly     | 306*(13)  |           |           |           |           |           |           |           |  |
| phPurgeOld          | 302*(13)  | 1158 (17) |           |           |           |           |           |           |  |
| phReopenDoc         | 303*(13)  | 742 (17)  |           |           |           |           |           |           |  |
| phRevert            | 300*(13)  | 333 (17)  |           |           |           |           |           |           |  |
| phSaveChanges       | 299*(13)  | 773 (17)  |           |           |           |           |           |           |  |
| phSpaceIsLow        | 304*(13)  | 2968 (16) |           |           |           |           |           |           |  |
| phSpoolPrintDialog  | 55*(36)   | 1389 (37) |           |           |           |           |           |           |  |
| phStylesTooBig      | 312*(13)  | 1096 (40) | 1098 (40) | 1155 (40) | 1161 (40) | 1471 (40) |           |           |  |
| phTooManyChars      | 308*(13)  | 855 (40)  |           |           |           |           |           |           |  |
| phUnimplemented     | 319*(13)  | 1053 (14) |           |           |           |           |           |           |  |
| phUnknownErr        | 296*(13)  | 220 (14)  |           |           |           |           |           |           |  |
| phUnsupportedROMs   | 305*(13)  | 406 (14)  |           |           |           |           |           |           |  |
| phWhichDoc          | 65*(36)   | 1997 (37) |           |           |           |           |           |           |  |
| picFrame            | 1863 (18) | 117 (23)  | 123 (23)  | 256 (23)  |           |           |           |           |  |
| PicHandle           | 394 ( 5)  | 418 ( 5)  | 1989 ( 6) | 1833 (18) | 117 (23)  | 123 (23)  | 256 (23)  | 263 (23)  |  |
| PictureTemplate     | 386 ( 5)  | 387*( 5)  | 1897 ( 6) | 1915 ( 6) |           |           |           |           |  |
| PictureTemplatePtr  | 386*( 5)  | 1887 ( 6) | 1910 ( 6) | 1915 ( 6) |           |           |           |           |  |
| pidleProc           | 1415*(37) |           |           |           |           |           |           |           |  |
| PinOnRect           | 310*(25)  | 674*(26)  | 682*(26)  |           |           |           |           |           |  |
| PinRect             | 323 (19)  |           |           |           |           |           |           |           |  |
| PInteger            | 3194 ( 6) | 616 (14)  | 686 (14)  | 2068 (16) | 132*(25)  | 133 (25)  | 65 (26)   | 119 (26)  |  |
|                     | 366 (26)  | 374 (26)  | 393 (26)  | 426 (26)  | 432 (26)  | 435 (26)  | 1149 (26) | 1157 (26) |  |
|                     | 165 (28)  | 567*(28)  | 577 (28)  | 578 (28)  | 579 (28)  | 580 (28)  | 813 (28)  | 899*(28)  |  |
|                     | 1128 (28) | 224 (30)  | 66 (33)   | 99 (33)   | 103 (33)  | 44 (32)   | 45 (32)   | 46 (32)   |  |
|                     | 54 (32)   | 55 (32)   | 68 (32)   | 73 (32)   | 78 (32)   | 81 (32)   | 88 (32)   | 91 (32)   |  |
| pINTEGER            | 399*(28)  |           |           |           |           |           |           |           |  |
| pInteger            | 113 (26)  | 114 (26)  |           |           |           |           |           |           |  |
| PinVRect            | 3207 (16) | 66*(42)   |           |           |           |           |           |           |  |
| pixelDelta          | 531*(20)  | 541*(20)  | 542 (20)  | 543 (20)  |           |           |           |           |  |
| pixelSize           | 173 ( 6)  |           |           |           |           |           |           |           |  |
| FixPatHandle        | 1773 ( 6) | 1787 ( 6) |           |           |           |           |           |           |  |
| PJMP                | 308*(30)  | 342 (30)  |           |           |           |           |           |           |  |
| PLongInt            | 169 (12)  | 204 (12)  | 281 (12)  | 566 (12)  | 569 (12)  | 2070 (16) | 134*(25)  | 135 (25)  |  |
|                     | 1046 (28) | 179 (33)  |           |           |           |           |           |           |  |
| PlotCIcon           | 1553 ( 6) |           |           |           |           |           |           |           |  |
| PlotIcon            | 1549 ( 6) |           |           |           |           |           |           |           |  |
| pMem                | 400 (28)  | 401*(28)  |           |           |           |           |           |           |  |
| PMenuCRsrc          | 16*(16)   | 17 (16)   | 361*(30)  | 362 (30)  |           |           |           |           |  |
| PMenuList           | 179 (30)  | 180*(30)  |           |           |           |           |           |           |  |
| pname               | 392*(26)  | 408 (26)  | 409*(26)  | 409 (26)  | 436*(26)  | 437 (26)  |           |           |  |
| pnLoc               | 186*(37)  |           |           |           |           |           |           |           |  |
| pnMode              | 188*(37)  |           |           |           |           |           |           |           |  |
| pnPat               | 189 (37)  |           |           |           |           |           |           |           |  |
| pnsSize             | 187*(37)  | 734 (37)  | 786 (37)  |           |           |           |           |           |  |
| PObj                | 18*(12)   | 94 (12)   | 106 (12)  | 132 (12)  | 196 (12)  | 326 (12)  | 393 (12)  | 523 (12)  |  |
| PObjNameTbl         | 53 (31)   | 54*(31)   |           |           |           |           |           |           |  |
| Point               | 469 ( 5)  | 470 ( 5)  | 588 ( 5)  | 589 ( 5)  | 2231 ( 6) | 2239 ( 6) | 2240 ( 6) | 2244 ( 6) |  |
|                     | 2406 ( 6) | 2475 ( 6) | 2938 ( 6) | 3061 ( 6) | 3062 ( 6) | 3317 ( 6) | 3334 ( 6) | 75 ( 9)   |  |
|                     | 223 ( 9)  | 224 ( 9)  | 319 ( 9)  | 619 (10)  | 620 (10)  | 1448 (10) | 1858 (10) | 622 (13)  |  |
|                     | 741 (13)  | 870 (13)  | 1348 (13) | 1494 (13) | 1498 (13) | 1596 (13) | 1610 (13) | 1622 (13) |  |
|                     | 1625 (13) | 1626 (13) | 1632 (13) | 1684 (13) | 1762 (13) | 1845 (13) | 1963 (13) | 1965 (13) |  |
|                     | 1989 (13) | 2022 (13) | 2047 (13) | 2068 (13) | 2099 (13) | 2100 (13) | 2175 (13) | 2176 (13) |  |
|                     | 2248 (13) | 2249 (13) | 2304 (13) | 2305 (13) | 2327 (13) | 2329 (13) | 2364 (13) | 2376 (13) |  |
|                     | 2698 (13) | 2764 (13) | 2774 (13) | 599 (14)  | 603 (14)  | 604 (14)  | 838 (14)  | 904 (14)  |  |
|                     | 262 (15)  | 333 (16)  | 415 (16)  | 1364 (16) | 1556 (16) | 1558 (16) | 2927 (16) | 2934 (16) |  |
|                     | 3001 (16) | 3098 (16) | 3105 (16) | 910 (17)  | 1400 (17) | 1406 (17) | 146 (18)  | 394 (18)  |  |
|                     | 717 (18)  | 802 (18)  | 803 (18)  | 841 (18)  | 876 (18)  | 1299 (18) | 1516 (18) | 1618 (18) |  |
|                     | 1625 (18) | 1641 (18) | 1793 (18) | 306 (19)  | 308 (19)  | 315 (19)  | 323 (19)  | 349 (19)  |  |
|                     | 686 (19)  | 784 (19)  | 848 (19)  | 866 (19)  | 945 (19)  | 948 (19)  | 428 (20)  | 112 (21)  |  |
|                     | 228 (21)  | 302 (21)  | 303 (21)  | 676 (21)  | 677 (21)  | 1048 (21) | 1049 (21) | 1341 (21) |  |
|                     | 1342 (21) | 56 (24)   | 141 (24)  | 310 (25)  | 452 (25)  | 470 (25)  | 674 (26)  | 886 (26)  |  |
|                     | 1172 (26) | 1182 (26) | 33 (30)   | 406 (30)  | 115 (36)  | 184 (36)  | 211 (36)  | 249 (36)  |  |
|                     | 255 (36)  | 320 (36)  | 186 (37)  | 187 (37)  | 436 (37)  | 500 (37)  | 501 (37)  | 774 (37)  |  |
|                     | 775 (37)  | 933 (37)  | 1092 (37) | 1224 (37) | 1229 (37) | 1247 (37) | 1254 (37) | 1452 (37) |  |
|                     | 1624 (37) | 1848 (37) | 1884 (37) | 2034 (37) | 195 (38)  | 196 (38)  | 214 (38)  | 178 (39)  |  |
|                     | 267 (40)  | 883 (40)  | 884 (40)  | 911 (40)  | 1294 (40) | 1296 (40) | 1297 (40) | 1299 (40) |  |
|                     | 31 (42)   | 33 (42)   |           |           |           |           |           |           |  |
| POINTER             | 586 (19)  | 1392 (37) | 1997 (37) |           |           |           |           |           |  |
| Pointer             | 246 ( 8)  | 597 (14)  | 1276 (16) | 1276 (16) | 1278 (16) | 1278 (16) | 120 (19)  | 123 (19)  |  |
|                     | 935 (26)  | 228 (30)  | 59 (41)   |           |           |           |           |           |  |
| PointerToHex        | 315*(25)  | 549*(26)  | 971 (26)  | 997 (26)  |           |           |           |           |  |
| pointInView         | 1258 (37) |           |           |           |           |           |           |           |  |
| pointInView         | 255*(36)  | 1247*(37) | 1263 (37) |           |           |           |           |           |  |

[illegible]



|                  |           |           |           |           |           |           |           |           |  |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| ProOpenPage      | 1595 (37) |           |           |           |           |           |           |           |  |
| protoObj         | 2988*(13) | 2991*(13) | 1155*(14) | 1161 (14) | 1171*(14) | 1179 (14) | 1182 (14) |           |  |
| PRPicFile        | 234 (37)  |           |           |           |           |           |           |           |  |
| PrSetError       | 80 (37)   | 83 (37)   | 639 (37)  | 1977 (37) |           |           |           |           |  |
| prStatus         | 230*(37)  | 234 (37)  |           |           |           |           |           |           |  |
| prStl            | 543 (37)  | 544 (37)  | 1725 (37) | 1756 (37) |           |           |           |           |  |
| PrStlDialog      | 1326 (37) |           |           |           |           |           |           |           |  |
| PRStr            | 834 ( 6)  | 917 ( 6)  | 1044 ( 6) | 1193 ( 6) | 2749 ( 6) | 2293 (10) | 262 (18)  | 203 (19)  |  |
|                  | 193 (21)  | 223*(25)  | 156 (40)  |           |           |           |           |           |  |
| PRunArray        | 83*( 9)   | 84 ( 9)   |           |           |           |           |           |           |  |
| PrValidate       | 2067 (37) |           |           |           |           |           |           |           |  |
| PScrapStuff      | 241 (16)  | 19 (23)   | 45 (23)   |           |           |           |           |           |  |
| PSeqHeader       | 893 (28)  | 894*(28)  |           |           |           |           |           |           |  |
| PSFTypeList      | 410 (16)  | 411*(16)  | 420 (16)  |           |           |           |           |           |  |
| PString8         | 140*(25)  | 141 (25)  |           |           |           |           |           |           |  |
| pt               | 146*(18)  | 148 (18)  | 148 (18)  | 945*(19)  | 957=(19)  | 958 (19)  | 959 (19)  | 960 (19)  |  |
|                  | 986=(19)  | 988 (19)  | 988 (19)  | 989 (19)  | 989 (19)  | 991 (19)  | 991 (19)  | 452*(25)  |  |
|                  | 464*(25)  | 470*(25)  | 482*(25)  | 1172*(26) | 1175 (26) | 1175 (26) | 1182*(26) | 1185 (26) |  |
|                  | 1270*(26) | 1273 (26) | 1273 (26) | 1280*(26) | 1283 (26) | 56*(42)   | 66*(42)   |           |  |
| pt1              | 41*(42)   |           |           |           |           |           |           |           |  |
| pt2              | 41*(42)   |           |           |           |           |           |           |           |  |
| Pt2Rect          | 87 (22)   |           |           |           |           |           |           |           |  |
| Pt2VRect         | 70*(42)   |           |           |           |           |           |           |           |  |
| PTBlock          | 10 (35)   | 11*(35)   | 109 (35)  |           |           |           |           |           |  |
| PTBlockPtr       | 10*(35)   | 101 (35)  | 109 (35)  | 130 (35)  |           |           |           |           |  |
| PTemplate        | 1223*(16) | 1224 (16) |           |           |           |           |           |           |  |
| PtInRect         | 253 (21)  |           |           |           |           |           |           |           |  |
| PtInRgn          | 1145 (10) | 1764 (10) | 2963 (10) | 2975 (10) | 2981 (10) | 3007 (10) | 3018 (16) | 148 (18)  |  |
|                  | 148 (18)  |           |           |           |           |           |           |           |  |
| PtInVRect        | 3300 (16) | 688 (18)  | 56*(42)   |           |           |           |           |           |  |
| PtIsVisible      | 146*(18)  | 148=(18)  |           |           |           |           |           |           |  |
| ptPtr            | 1706*( 6) | 1711=( 6) | 1713 ( 6) |           |           |           |           |           |  |
| Ptr              | 39 ( 1)   | 87 ( 1)   | 56 ( 2)   | 356 ( 2)  | 73 ( 5)   | 77 ( 5)   | 79 ( 5)   | 124 ( 5)  |  |
|                  | 147 ( 5)  | 151 ( 5)  | 153 ( 5)  | 177 ( 5)  | 181 ( 5)  | 183 ( 5)  | 217 ( 5)  | 221 ( 5)  |  |
|                  | 223 ( 5)  | 260 ( 5)  | 264 ( 5)  | 266 ( 5)  | 285 ( 5)  | 312 ( 5)  | 316 ( 5)  | 318 ( 5)  |  |
|                  | 331 ( 5)  | 358 ( 5)  | 362 ( 5)  | 364 ( 5)  | 377 ( 5)  | 403 ( 5)  | 407 ( 5)  | 409 ( 5)  |  |
|                  | 422 ( 5)  | 452 ( 5)  | 456 ( 5)  | 458 ( 5)  | 491 ( 5)  | 519 ( 5)  | 523 ( 5)  | 525 ( 5)  |  |
|                  | 536 ( 5)  | 547 ( 5)  | 573 ( 5)  | 577 ( 5)  | 579 ( 5)  | 599 ( 5)  | 624 ( 5)  | 649 ( 5)  |  |
|                  | 653 ( 5)  | 655 ( 5)  | 666 ( 5)  | 689 ( 5)  | 298 ( 6)  | 345 ( 6)  | 367 ( 6)  | 682 ( 6)  |  |
|                  | 682 ( 6)  | 755 ( 6)  | 796 ( 6)  | 819 ( 6)  | 844 ( 6)  | 878 ( 6)  | 901 ( 6)  | 924 ( 6)  |  |
|                  | 1006 ( 6) | 1028 ( 6) | 1051 ( 6) | 1158 ( 6) | 1178 ( 6) | 1200 ( 6) | 1287 ( 6) | 1368 ( 6) |  |
|                  | 1433 ( 6) | 1481 ( 6) | 1509 ( 6) | 1591 ( 6) | 1657 ( 6) | 1703 ( 6) | 1728 ( 6) | 1811 ( 6) |  |
|                  | 1870 ( 6) | 1907 ( 6) | 1928 ( 6) | 2003 ( 6) | 2080 ( 6) | 2139 ( 6) | 2165 ( 6) | 2350 ( 6) |  |
|                  | 2528 ( 6) | 2562 ( 6) | 2708 ( 6) | 2731 ( 6) | 2757 ( 6) | 2819 ( 6) | 2846 ( 6) | 2903 ( 6) |  |
|                  | 2951 ( 6) | 2974 ( 6) | 2995 ( 6) | 3135 ( 6) | 3167 ( 6) | 3167 ( 6) | 3175 ( 6) | 3415 ( 6) |  |
|                  | 3441 ( 6) | 3464 ( 6) | 3541 ( 6) | 214 ( 7)  | 62 ( 8)   | 194 ( 9)  | 199 ( 9)  | 202 ( 9)  |  |
|                  | 399 ( 9)  | 452 ( 9)  | 457 ( 9)  | 460 ( 9)  | 496 ( 9)  | 543 ( 9)  | 624 ( 9)  | 661 ( 9)  |  |
|                  | 699 ( 9)  | 722 ( 9)  | 745 ( 9)  | 768 ( 9)  | 140 (10)  | 240 (10)  | 277 (10)  | 842 (10)  |  |
|                  | 891 (10)  | 902 (10)  | 939 (10)  | 988 (10)  | 999 (10)  | 1570 (10) | 1584 (10) | 1670 (10) |  |
|                  | 1684 (10) | 2139 (10) | 2234 (10) | 2270 (10) | 2302 (10) | 2406 (10) | 2472 (10) | 2762 (10) |  |
|                  | 2833 (10) | 3059 (10) | 3094 (10) | 3127 (10) | 3162 (10) | 248 (11)  | 208 (12)  | 457 (12)  |  |
|                  | 621 (12)  | 639 (12)  | 608 (13)  | 652 (13)  | 698 (13)  | 921 (13)  | 1160 (13) | 1482 (13) |  |
|                  | 1484 (13) | 1486 (13) | 1732 (13) | 1776 (13) | 1780 (13) | 1782 (13) | 1856 (13) | 1936 (13) |  |
|                  | 1940 (13) | 1942 (13) | 2032 (13) | 2078 (13) | 2082 (13) | 2084 (13) | 2139 (13) | 2159 (13) |  |
|                  | 2161 (13) | 2163 (13) | 2206 (13) | 2237 (13) | 2241 (13) | 2243 (13) | 2271 (13) | 2291 (13) |  |
|                  | 2295 (13) | 2297 (13) | 2315 (13) | 2347 (13) | 2467 (13) | 2565 (13) | 2581 (13) | 2927 (13) |  |
|                  | 2936 (13) | 2943 (13) | 40 (15)   | 57 (15)   | 180 (15)  | 907 (16)  | 1210 (16) | 1485 (17) |  |
|                  | 31 (18)   | 41 (18)   | 97 (18)   | 128 (18)  | 137 (18)  | 219 (18)  | 239 (18)  | 275 (18)  |  |
|                  | 1903 (18) | 83 (19)   | 171 (19)  | 214 (19)  | 1024 (19) | 46 (20)   | 80 (20)   | 111 (20)  |  |
|                  | 295 (20)  | 127 (21)  | 162 (21)  | 204 (21)  | 503 (21)  | 552 (21)  | 567 (21)  | 579 (21)  |  |
|                  | 923 (21)  | 962 (21)  | 991 (21)  | 1017 (21) | 1181 (21) | 1244 (21) | 1273 (21) | 1285 (21) |  |
|                  | 1431 (21) | 120 (22)  | 42 (23)   | 135 (23)  | 39 (24)   | 220 (25)  | 223 (25)  | 240 (25)  |  |
|                  | 245 (25)  | 303 (25)  | 350 (25)  | 357 (25)  | 50 (26)   | 166 (26)  | 186 (26)  | 526 (26)  |  |
|                  | 859 (26)  | 881 (26)  | 1096 (26) | 1322 (26) | 172 (28)  | 816 (28)  | 1049 (28) | 1137 (28) |  |
|                  | 311 (30)  | 97 (31)   | 115 (31)  | 34 (33)   | 152 (32)  | 255 (32)  | 290 (32)  | 63 (34)   |  |
|                  | 78 (34)   | 86 (34)   | 96 (34)   | 13 (35)   | 22 (35)   | 32 (35)   | 64 (35)   | 72 (35)   |  |
|                  | 99 (35)   | 144 (35)  | 178 (35)  | 467 (36)  | 504 (36)  | 144 (37)  | 146 (37)  | 875 (37)  |  |
|                  | 2150 (37) | 160 (38)  | 165 (38)  | 167 (38)  | 357 (38)  | 449 (38)  | 533 (38)  | 587 (38)  |  |
|                  | 70 (40)   | 127 (40)  | 164 (40)  | 499 (40)  | 1634 (40) | 162 (41)  | 611 (41)  | 679 (41)  |  |
|                  | 733 (41)  | 756 (41)  | 938 (41)  |           |           |           |           |           |  |
| PtrAndHand       | 679 (41)  |           |           |           |           |           |           |           |  |
| PtrHandle        | 958*(28)  | 962 (28)  | 966 (28)  | 970 (28)  |           |           |           |           |  |
| ptrToObject      | 221*(32)  | 225=(32)  | 226 (32)  |           |           |           |           |           |  |
| PtrToHand        | 1443 (40) |           |           |           |           |           |           |           |  |
| ptrToRgnHandle   | 557*(19)  | 586=(19)  | 587 (19)  |           |           |           |           |           |  |
| PtrToXHand       | 1150 (40) |           |           |           |           |           |           |           |  |
| PtToVPt          | 3255 (16) | 3278 (16) | 3345 (16) | 1518 (18) | 31*(42)   |           |           |           |  |
| PTypeList        | 551 (13)  | 552*(13)  |           |           |           |           |           |           |  |
| pTypeList        | 420*(16)  | 433=(16)  | 433 (16)  | 439=(16)  | 450 (16)  |           |           |           |  |
| PublicizeFailed  | 199*(16)  | 226 (16)  |           |           |           |           |           |           |  |
| PurgeMem         | 260 (28)  |           |           |           |           |           |           |           |  |
| pushButProc      | 785 ( 6)  | 807 ( 6)  |           |           |           |           |           |           |  |
| PutDeskScrapData | 2895*(13) | 1117*(14) | 1128=(14) | 1872 (18) | 1603 (40) | 1610 (40) |           |           |  |
| putDlgID         | 1405 (17) |           |           |           |           |           |           |           |  |
| PutScrap         | 1122 (14) |           |           |           |           |           |           |           |  |
| PWStateData      | 869*(19)  | 870 (19)  |           |           |           |           |           |           |  |
| PX               | 41*(39)   |           |           |           |           |           |           |           |  |
| Px               | 42 (39)   |           |           |           |           |           |           |           |  |
| -Q-              |           |           |           |           |           |           |           |           |  |
| q                | 29*( 4)   | 134 ( 4)  | 149 ( 4)  | 183 ( 4)  | 218 ( 4)  | 240 ( 4)  |           |           |  |
| QDByte           | 603 (30)  |           |           |           |           |           |           |           |  |
| qdEndPt          | 775*(37)  | 787=(37)  | 792 (37)  | 792 (37)  | 810 (37)  | 818 (37)  |           |           |  |
| qdExtent         | 1647*(13) | 1180*(18) | 1182 (18) | 1183 (18) | 1834*(18) | 1840 (18) | 1848 (18) | 1854 (18) |  |
|                  | 1856 (18) | 319*(21)  | 324 (21)  | 325 (21)  | 694*(21)  | 712 (21)  | 713 (21)  | 914*(40)  |  |
|                  | 921 (40)  | 922 (40)  |           |           |           |           |           |           |  |
| qdFrame          | 1003*(18) | 1032 (18) | 1047 (18) |           |           |           |           |           |  |
| qdPoint          | 1494*(13) | 1516*(18) | 1518 (18) |           |           |           |           |           |  |
| QDPtr            | 503 (40)  | 508 (40)  | 510 (40)  |           |           |           |           |           |  |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| qdRect            | 1496*(13) | 1500*(13) | 1526*(18) | 1528 (18) | 1803*(18) | 1806 (18) |           |           |  |
| qdStartPt         | 774*(37)  | 785-(37)  | 791 (37)  | 791 (37)  | 806 (37)  | 814 (37)  |           |           |  |
| QDToViewPt        | 1456 (10) | 1494*(13) | 1612 (16) | 3252 (16) | 3276 (16) | 3342 (16) | 1516*(18) | 197 (39)  |  |
| QDToViewRect      | 1496*(13) | 1026 (18) | 1526*(18) | 753 (37)  | 1330 (40) |           |           |           |  |
| QDToViewRect      | 702 (10)  |           |           |           |           |           |           |           |  |
| QElem             | 29 (4)    |           |           |           |           |           |           |           |  |
| qType             | 136-(4)   |           |           |           |           |           |           |           |  |
| QuickDraw         | 14*(1)    | 58*(3)    | 14*(5)    | 136*(7)   | 25*(9)    | 17*(11)   | 45*(13)   | 44*(25)   |  |
|                   | 130*(27)  | 110*(29)  | 15*(31)   | 53*(34)   | 14*(36)   | 21*(38)   | 14*(42)   |           |  |
| -R-               |           |           |           |           |           |           |           |           |  |
| r                 | 1586*(13) | 1608*(13) | 2979*(13) | 1135*(14) | 1139 (14) | 1145 (14) | 3008*(16) | 3054 (16) |  |
|                   | 3058 (16) | 155*(18)  | 163 (18)  | 165-(18)  | 579*(18)  | 587 (18)  | 1342*(18) | 1346 (18) |  |
|                   | 1347 (18) | 1357*(18) | 1362 (18) | 1363 (18) | 1364 (18) | 1557*(18) | 1580 (18) | 1581 (18) |  |
|                   | 1582 (18) | 1585 (18) | 1586 (18) | 1590 (18) | 1767*(18) | 1772 (18) | 1773 (18) | 1774 (18) |  |
|                   | 472*(19)  | 478-(19)  | 479 (19)  | 479 (19)  | 480 (19)  | 480 (19)  | 487 (19)  | 505*(19)  |  |
|                   | 819*(19)  | 829 (19)  | 831 (19)  | 832 (19)  | 833 (19)  | 835 (19)  | 836 (19)  | 483*(20)  |  |
|                   | 490 (20)  | 495 (20)  | 503 (20)  | 509 (20)  | 83*(22)   | 87 (22)   | 88 (22)   | 288*(25)  |  |
|                   | 458*(25)  | 467*(25)  | 476*(25)  | 485*(25)  | 468*(26)  | 471 (26)  | 471 (26)  | 1220*(26) |  |
|                   | 1223 (26) | 1225 (26) | 1232*(26) | 1235 (26) | 1292*(26) | 1295 (26) | 1297 (26) | 1304*(26) |  |
|                   | 1307 (26) | 1261*(40) | 1268 (40) | 1269 (40) | 1270 (40) | 1272 (40) | 1273 (40) | 50*(42)   |  |
|                   | 52*(42)   | 54*(42)   | 56*(42)   | 60*(42)   | 64*(42)   | 66*(42)   |           |           |  |
| radioButProc      | 994 (6)   | 1017 (6)  |           |           |           |           |           |           |  |
| RadioTemplate     | 202 (5)   | 203*(5)   | 1018 (6)  | 1040 (6)  |           |           |           |           |  |
| RadioTemplatePtr  | 202*(5)   | 1016 (6)  | 1032 (6)  | 1039 (6)  |           |           |           |           |  |
| rangeEnd          | 40*(38)   | 47*(38)   | 317*(38)  | 1464*(40) | 1468 (40) |           |           |           |  |
| rangeStart        | 40*(38)   | 47*(38)   | 317*(38)  | 1464*(40) | 1468 (40) |           |           |           |  |
| ranOutOfSpace     | 327*(36)  | 111*(37)  | 127-(37)  | 1119*(37) | 1143-(37) | 1157 (37) | 1175 (37) | 1183 (37) |  |
|                   | 1185 (37) | 1448*(37) | 1466-(37) | 1508 (37) | 1512 (37) |           |           |           |  |
| rdPtr             | 1032*(6)  | 1039-(6)  | 1042 (6)  | 1044 (6)  |           |           |           |           |  |
| react             | 1318*(37) | 1326-(37) | 1331-(37) | 1336 (37) | 1350 (37) | 1361 (37) |           |           |  |
| ReadFromDeskScrap | 1040*(13) | 400 (16)  | 1895 (16) | 2486*(16) |           |           |           |           |  |
| ReadFromFile      | 1213*(13) | 2221 (16) | 2450 (16) | 799*(17)  | 996 (17)  |           |           |           |  |
| ReadInteger       | 49 (8)    | 100 (8)   | 128 (8)   | 445*(25)  | 690*(26)  | 700-(26)  |           |           |  |
| ReadLn            | 698 (26)  | 717 (26)  | 351 (28)  | 358 (28)  | 365 (28)  |           |           |           |  |
| ReadYesNo         | 448*(25)  | 709*(26)  | 719-(26)  | 80 (33)   | 188 (33)  |           |           |           |  |
| RealInitPrinting  | 173*(37)  | 219 (37)  |           |           |           |           |           |           |  |
| ReallocHandle     | 210 (28)  | 276 (28)  |           |           |           |           |           |           |  |
| realText          | 530*(5)   | 593*(5)   | 1253*(6)  | 1274-(6)  | 1275 (6)  | 1277-(6)  | 1282 (6)  | 1284 (6)  |  |
|                   | 1286 (6)  | 1287 (6)  | 1288 (6)  | 1289 (6)  | 1291 (6)  | 2779*(6)  | 2786-(6)  | 2787 (6)  |  |
|                   | 2789-(6)  | 2794 (6)  | 2808*(6)  | 2813 (6)  | 2818 (6)  | 2819 (6)  | 2820 (6)  | 2821 (6)  |  |
|                   | 2823 (6)  | 3102*(6)  | 3105-(6)  |           |           |           |           |           |  |
| reason            | 759*(17)  | 769 (17)  | 771 (17)  |           |           |           |           |           |  |
| RecalcText        | 683*(5)   | 2599 (6)  | 2636*(6)  | 2638 (6)  | 3294 (6)  | 294*(38)  | 357 (40)  | 1247*(40) |  |
|                   | 1276 (40) | 1425 (40) | 1454 (40) | 1509 (40) | 557 (41)  |           |           |           |  |
| recovErr          | 174*(14)  | 211-(14)  | 213-(14)  | 215 (14)  |           |           |           |           |  |
| recovery          | 175*(14)  | 215 (14)  | 217 (14)  |           |           |           |           |           |  |
| RECT              | 1216 (16) |           |           |           |           |           |           |           |  |
| Rect              | 271 (5)   | 323 (5)   | 369 (5)   | 414 (5)   | 465 (5)   | 467 (5)   | 472 (5)   | 474 (5)   |  |
|                   | 476 (5)   | 532 (5)   | 536 (5)   | 595 (5)   | 599 (5)   | 145 (6)   | 799 (6)   | 881 (6)   |  |
|                   | 1009 (6)  | 1245 (6)  | 1252 (6)  | 1531 (6)  | 1534 (6)  | 1750 (6)  | 1753 (6)  | 1950 (6)  |  |
|                   | 1953 (6)  | 2207 (6)  | 2223 (6)  | 2246 (6)  | 2247 (6)  | 2291 (6)  | 2294 (6)  | 2335 (6)  |  |
|                   | 2341 (6)  | 2358 (6)  | 2370 (6)  | 2371 (6)  | 2462 (6)  | 2803 (6)  | 2806 (6)  | 2846 (6)  |  |
|                   | 2879 (6)  | 2928 (6)  | 3113 (6)  | 3116 (6)  | 3175 (6)  | 3285 (6)  | 227 (9)   | 235 (9)   |  |
|                   | 236 (9)   | 252 (9)   | 255 (9)   | 479 (9)   | 329 (10)  | 339 (10)  | 492 (10)  | 544 (10)  |  |
|                   | 545 (10)  | 567 (10)  | 568 (10)  | 686 (10)  | 691 (10)  | 767 (10)  | 823 (10)  | 850 (10)  |  |
|                   | 947 (10)  | 1141 (10) | 1424 (10) | 1987 (10) | 1988 (10) | 2315 (10) | 2872 (10) | 2873 (10) |  |
|                   | 2874 (10) | 2875 (10) | 2936 (10) | 1496 (13) | 1500 (13) | 1586 (13) | 1592 (13) | 1596 (13) |  |
|                   | 1608 (13) | 1647 (13) | 1651 (13) | 1695 (13) | 1724 (13) | 1753 (13) | 1907 (13) | 1908 (13) |  |
|                   | 1981 (13) | 2018 (13) | 2053 (13) | 2069 (13) | 2093 (13) | 2102 (13) | 2114 (13) | 2170 (13) |  |
|                   | 2178 (13) | 2307 (13) | 2401 (13) | 2572 (13) | 2776 (13) | 2778 (13) | 2780 (13) | 2814 (13) |  |
|                   | 2979 (13) | 32 (14)   | 479 (14)  | 1135 (14) | 3007 (16) | 3008 (16) | 155 (18)  | 394 (18)  |  |
|                   | 579 (18)  | 756 (18)  | 863 (18)  | 874 (18)  | 886 (18)  | 1003 (18) | 1004 (18) | 1180 (18) |  |
|                   | 1209 (18) | 1342 (18) | 1357 (18) | 1508 (18) | 1526 (18) | 1557 (18) | 1767 (18) | 1803 (18) |  |
|                   | 1834 (18) | 86 (19)   | 307 (19)  | 350 (19)  | 444 (19)  | 472 (19)  | 505 (19)  | 553 (19)  |  |
|                   | 554 (19)  | 626 (19)  | 787 (19)  | 819 (19)  | 852 (19)  | 944 (19)  | 483 (20)  | 250 (21)  |  |
|                   | 260 (21)  | 276 (21)  | 317 (21)  | 319 (21)  | 354 (21)  | 438 (21)  | 536 (21)  | 623 (21)  |  |
|                   | 690 (21)  | 694 (21)  | 781 (21)  | 965 (21)  | 1384 (21) | 1387 (21) | 83 (22)   | 78 (23)   |  |
|                   | 229 (23)  | 232 (23)  | 90 (24)   | 288 (25)  | 310 (25)  | 320 (25)  | 458 (25)  | 476 (25)  |  |
|                   | 468 (26)  | 674 (26)  | 727 (26)  | 887 (26)  | 1220 (26) | 1232 (26) | 33 (30)   | 405 (30)  |  |
|                   | 83 (36)   | 84 (36)   | 85 (36)   | 86 (36)   | 365 (36)  | 424 (36)  | 518 (36)  | 56 (37)   |  |
|                   | 353 (37)  | 712 (37)  | 914 (37)  | 931 (37)  | 932 (37)  | 1046 (37) | 1376 (37) | 1626 (37) |  |
|                   | 1833 (37) | 78 (38)   | 101 (38)  | 152 (38)  | 218 (38)  | 270 (38)  | 330 (38)  | 180 (39)  |  |
|                   | 16 (40)   | 391 (40)  | 677 (40)  | 914 (40)  | 982 (40)  | 1205 (40) | 1261 (40) | 1298 (40) |  |
|                   | 1300 (40) | 1553 (40) | 1576 (40) | 352 (41)  | 46 (42)   | 48 (42)   |           |           |  |
| rectA             | 62*(42)   |           |           |           |           |           |           |           |  |
| rectB             | 62*(42)   |           |           |           |           |           |           |           |  |
| RectInRgn         | 1145 (14) | 898 (18)  | 593 (19)  |           |           |           |           |           |  |
| RectIsVisible     | 2979*(13) | 1135*(14) | 1139-(14) | 1143-(14) | 1145-(14) |           |           |           |  |
| RectRgn           | 501 (10)  | 526 (10)  | 872 (10)  | 969 (10)  | 2904 (10) | 2920 (10) | 2951 (10) | 2971 (10) |  |
|                   | 3010 (10) | 3058 (16) | 163 (18)  | 587 (18)  | 1582 (18) | 1590 (18) | 922 (40)  |           |  |
| RectsNest         | 320*(25)  | 727*(26)  | 730-(26)  | 1317 (40) |           |           |           |           |  |
| rectToClip        | 931*(37)  | 960 (37)  | 961 (37)  | 961 (37)  | 962 (37)  |           |           |           |  |
| rectToClipTo      | 914*(37)  | 916-(37)  | 921 (37)  |           |           |           |           |           |  |
| rectToReveal      | 1610*(13) | 1845*(13) | 1613*(18) | 1616 (18) | 1617 (18) | 1618 (18) | 1625*(18) | 1630 (18) |  |
|                   | 1631 (18) | 428*(20)  | 444 (20)  | 446 (20)  |           |           |           |           |  |
| RectToVRect       | 1528 (18) | 66 (20)   | 46*(42)   |           |           |           |           |           |  |
| red               | 97-(6)    | 121 (6)   | 234 (6)   | 237 (6)   | 342*(25)  | 318-(26)  | 792 (26)  | 829*(26)  |  |
|                   | 832-(26)  | 832 (26)  | 1019 (26) | 1021 (26) | 1025 (26) |           |           |           |  |
| redColor          | 132 (6)   | 803 (26)  |           |           |           |           |           |           |  |
| RedoIt            | 2475*(13) | 810 (16)  | 62*(22)   | 500*(36)  | 2177*(37) | 411*(38)  | 528*(38)  | 299*(41)  |  |
|                   | 599*(41)  |           |           |           |           |           |           |           |  |
| RedoPageBreaks    | 2384*(13) | 815 (18)  | 193*(24)  | 377*(36)  | 1617*(37) |           |           |           |  |
| redraw            | 190*(5)   | 192*(5)   | 194*(5)   | 230*(5)   | 232*(5)   | 234*(5)   | 281*(5)   | 327*(5)   |  |
|                   | 373*(5)   | 418*(5)   | 484*(5)   | 487*(5)   | 541*(5)   | 543*(5)   | 610*(5)   | 612*(5)   |  |
|                   | 614*(5)   | 660*(5)   | 956*(5)   | 958 (6)   | 965*(6)   | 967 (6)   | 974*(6)   | 977 (6)   |  |
|                   | 1085*(6)  | 1087 (6)  | 1094*(6)  | 1096 (6)  | 1103*(6)  | 1106 (6)  | 1348*(6)  | 1358 (6)  |  |
|                   | 1577*(6)  | 1581 (6)  | 1797*(6)  | 1801 (6)  | 1989*(6)  | 1993 (6)  | 2460*(6)  | 2472 (6)  |  |
|                   | 2484*(6)  | 2518 (6)  | 2866*(6)  | 2869 (6)  | 2877*(6)  | 2888 (6)  | 3256*(6)  | 3260 (6)  |  |

|                           |       |      |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
|---------------------------|-------|------|-------|------|-------|------|-------|------|-------|------|-------|------|-------|------|-------|------|
|                           | 3267* | ( 6) | 3271  | ( 6) | 3281* | ( 6) | 3295  | ( 6) | 3304  | ( 6) | 3493* | ( 6) | 3501  | ( 6) | 359*  | ( 9) |
|                           | 1852* | (10) | 1871  | (10) | 1611* | (13) | 1613* | (13) | 1615* | (13) | 1653* | (13) | 1657* | (13) | 1833* | (13) |
|                           | 1839* | (13) | 1841* | (13) | 1846* | (13) | 2000* | (13) | 2097* | (13) | 2108* | (13) | 2110* | (13) | 2114* | (13) |
|                           | 2116* | (13) | 2118* | (13) | 2173* | (13) | 2188* | (13) | 2192* | (13) | 2194* | (13) | 2196* | (13) | 2198* | (13) |
|                           | 2200* | (13) | 2202* | (13) | 2257* | (13) | 2259* | (13) | 2261* | (13) | 2263* | (13) | 1610* | (18) | 1618  | (18) |
|                           | 1626* | (18) | 1631  | (18) | 1639* | (18) | 1641  | (18) | 1648* | (18) | 1650- | (18) | 1650  | (18) | 1651  | (18) |
|                           | 1782* | (18) | 1785  | (18) | 925*  | (19) | 928   | (19) | 254*  | (20) | 270   | (20) | 283   | (20) | 429*  | (20) |
|                           | 450   | (20) | 457*  | (20) | 471   | (20) | 592*  | (20) | 595   | (20) | 288*  | (21) | 293   | (21) | 364*  | (21) |
|                           | 369   | (21) | 378*  | (21) | 385   | (21) | 393*  | (21) | 396   | (21) | 404*  | (21) | 407   | (21) | 438*  | (21) |
|                           | 443   | (21) | 659*  | (21) | 669   | (21) | 760*  | (21) | 769   | (21) | 798*  | (21) | 809   | (21) | 816*  | (21) |
|                           | 827   | (21) | 834*  | (21) | 845   | (21) | 852*  | (21) | 863   | (21) | 870*  | (21) | 883   | (21) | 890*  | (21) |
|                           | 897   | (21) | 1108* | (21) | 1120  | (21) | 1128* | (21) | 1134  | (21) | 1142* | (21) | 1149  | (21) | 1157* | (21) |
|                           | 1161  | (21) | 41*   | (38) | 246*  | (38) | 273*  | (38) | 301*  | (38) | 307*  | (38) | 347*  | (40) | 355   | (40) |
|                           | 1348* | (40) | 1352  | (40) | 1360* | (40) | 1379  | (40) | 1426  | (40) | 1427  | (40) | 1574* | (40) | 1587  | (40) |
| refcon                    | 1220* | (16) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| RegainControl             | 1049* | (13) | 1676  | (16) | 2421  | (16) | 2518* | (16) |       |      |       |      |       |      |       |      |
| RegisterStdType           | 31    | ( 6) | 32    | ( 6) | 34    | ( 6) | 35    | ( 6) | 36    | ( 6) | 37    | ( 6) | 38    | ( 6) | 39    | ( 6) |
|                           | 40    | ( 6) | 41    | ( 6) | 42    | ( 6) | 43    | ( 6) | 44    | ( 6) | 2991* | (13) | 693   | (14) | 694   | (14) |
|                           | 695   | (14) | 1171* | (14) | 27    | (39) |       |      |       |      |       |      |       |      |       |      |
| RegisterType              | 33    | ( 6) | 2988* | (13) | 696   | (14) | 1155* | (14) | 1182  | (14) |       |      |       |      |       |      |
| regs                      | 152*  | ( 7) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| ReleaseIcon               | 325*  | ( 5) | 1523  | ( 6) | 1544  | ( 6) | 1563* | ( 6) | 1579  | ( 6) |       |      |       |      |       |      |
| ReleaseLabel              | 277*  | ( 5) | 1214  | ( 6) | 1313* | ( 6) | 1351  | ( 6) |       |      |       |      |       |      |       |      |
| ReleasePattern            | 371*  | ( 5) | 1742  | ( 6) | 1764  | ( 6) | 1783* | ( 6) | 1799  | ( 6) |       |      |       |      |       |      |
| ReleasePicture            | 416*  | ( 5) | 1942  | ( 6) | 1962  | ( 6) | 1979* | ( 6) | 1991  | ( 6) |       |      |       |      |       |      |
| ReleasePopup              | 482*  | ( 5) | 2179  | ( 6) | 2440* | ( 6) | 2499  | ( 6) |       |      |       |      |       |      |       |      |
| ReleaseResource           | 45    | (16) | 488   | (28) | 500   | (28) | 828   | (28) | 380   | (30) |       |      |       |      |       |      |
| ReleaseText               | 539*  | ( 5) | 2771  | ( 6) | 2855* | ( 6) | 2884  | ( 6) |       |      |       |      |       |      |       |      |
| relOrigin                 | 1002* | (18) | 1036  | (18) | 1041  | (18) | 1045  | (18) | 1046  | (18) | 1047  | (18) | 1047  | (18) |       |      |
| RemHandle                 | 273*  | (27) | 935*  | (28) |       |      |       |      |       |      |       |      |       |      |       |      |
| RemoveAdditions           | 438*  | (38) | 188*  | (41) | 283   | (41) |       |      |       |      |       |      |       |      |       |      |
| RemoveDeletions           | 234*  | (11) | 253   | (12) | 550*  | (12) |       |      |       |      |       |      |       |      |       |      |
| RemoveKeyAt               | 83*   | ( 1) | 295*  | ( 2) |       |      |       |      |       |      |       |      |       |      |       |      |
| RemoveObjectFromInspector | 20*   | (33) | 253   | (33) |       |      |       |      |       |      |       |      |       |      |       |      |
| RemoveSubView             | 2585  | ( 6) | 1791* | (13) | 2754  | (16) | 293   | (18) | 1536* | (18) | 386*  | (20) | 390   | (20) |       |      |
| RemoveSubview             | 1530* | (13) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| RemoveValueAt             | 81*   | ( 1) | 267*  | ( 2) |       |      |       |      |       |      |       |      |       |      |       |      |
| RemoveView                | 2751* | (16) | 2764  | (16) |       |      |       |      |       |      |       |      |       |      |       |      |
| Rename                    | 1370  | (17) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| ReplaceOnce               | 672*  | ( 6) | 690   | ( 6) |       |      |       |      |       |      |       |      |       |      |       |      |
| ReplaceText               | 116*  | ( 5) | 667*  | ( 6) | 1282  | ( 6) | 2794  | ( 6) |       |      |       |      |       |      |       |      |
| reply                     | 1233* | (14) | 1235- | (14) | 422*  | (16) | 450   | (16) | 454   | (16) | 455   | (16) | 457   | (16) | 458   | (16) |
|                           | 459   | (16) | 460   | (16) | 907*  | (17) | 928   | (17) | 930   | (17) | 932   | (17) | 933   | (17) |       |      |
| ReportCurrent             | 279*  | ( 5) | 1324* | ( 6) | 1339- | ( 6) | 1341- | ( 6) |       |      |       |      |       |      |       |      |
| ReportEvent               | 784*  | (13) | 1411  | (16) | 2532* | (16) |       |      |       |      |       |      |       |      |       |      |
| ReportMouseDown           | 787*  | (13) | 1570  | (16) | 2645* | (16) |       |      |       |      |       |      |       |      |       |      |
| RequestFileName           | 1284* | (13) | 901*  | (17) | 1087  | (17) |       |      |       |      |       |      |       |      |       |      |
| ResError                  | 120   | ( 8) | 2090  | (16) | 2101  | (16) | 40    | (26) | 654   | (26) |       |      |       |      |       |      |
| reserveSize               | 620*  | (28) | 649-  | (28) | 654   | (28) | 658   | (28) | 663   | (28) | 677-  | (28) | 683   | (28) |       |      |
| Reset                     | 2388* | (13) | 979   | (17) | 202*  | (24) | 380*  | (36) | 1678* | (37) | 1819  | (37) | 2075  | (37) |       |      |
| ResetAlertStage           | 226   | (14) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| ResetBusyCursor           | 87*   | ( 3) | 154   | ( 4) | 155   | ( 4) | 267*  | ( 4) |       |      |       |      |       |      |       |      |
| ResetPrintHandler         | 976*  | (17) | 1000  | (17) |       |      |       |      |       |      |       |      |       |      |       |      |
| ResetRadios               | 1227* | ( 6) | 1237  | ( 6) |       |      |       |      |       |      |       |      |       |      |       |      |
| resNotFound               | 1213  | (37) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| resIndex                  | 796*  | (28) | 807-  | (28) | 809   | (28) |       |      |       |      |       |      |       |      |       |      |
| resizable                 | 1869* | (13) | 128   | (19) | 189-  | (19) |       |      |       |      |       |      |       |      |       |      |
| Resize                    | 606*  | ( 5) | 2199  | ( 6) | 2592  | ( 6) | 3231* | ( 6) | 3234  | ( 6) | 3236  | ( 6) | 600*  | ( 9) | 2693* | (10) |
|                           | 2695  | (10) | 1566* | (13) | 1809* | (13) | 1967* | (13) | 2120* | (13) | 2190* | (13) | 884   | (14) | 961   | (14) |
|                           | 1032  | (14) | 385   | (18) | 1551* | (18) | 1729  | (18) | 805   | (19) | 816*  | (19) | 839   | (19) | 858   | (19) |
|                           | 1014  | (19) | 172   | (20) | 397*  | (20) | 415   | (20) | 385   | (21) | 424*  | (21) | 429   | (21) | 443   | (21) |
|                           | 777*  | (21) | 791   | (21) | 243*  | (38) | 1258* | (40) | 1265  | (40) |       |      |       |      |       |      |
| ResizeByUser              | 1965* | (13) | 1624  | (16) | 848*  | (19) |       |      |       |      |       |      |       |      |       |      |
| ResizeCMgrControl         | 779*  | (21) | 790   | (21) |       |      |       |      |       |      |       |      |       |      |       |      |
| resizeLimits              | 852*  | (19) | 855-  | (19) | 856   | (19) |       |      |       |      |       |      |       |      |       |      |
| resLoad                   | 372*  | (25) | 1144* | (26) | 1153  | (26) | 1158  | (26) |       |      |       |      |       |      |       |      |
| resNotFound               | 2093  | (16) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| resourceID                | 2910* | (13) | 714*  | (14) | 720*  | (14) | 739   | (14) |       |      |       |      |       |      |       |      |
| resourceId                | 771   | (14) | 774   | (14) |       |      |       |      |       |      |       |      |       |      |       |      |
| RestartEdit               | 608*  | ( 5) | 412   | ( 6) | 3244* | ( 6) |       |      |       |      |       |      |       |      |       |      |
| RestoreAS                 | 275   | ( 4) | 301   | ( 4) | 329   | ( 4) | 114   | (14) | 698   | (28) | 783   | (28) |       |      |       |      |
| RestoreSelection          | 421*  | (38) | 203*  | (41) | 285   | (41) | 303   | (41) |       |      |       |      |       |      |       |      |
| restWidth                 | 407*  | (40) | 503-  | (40) | 510-  | (40) | 518   | (40) | 521   | (40) |       |      |       |      |       |      |
| ResType                   | 1028  | (13) | 1663  | (13) | 1665  | (13) | 2742  | (13) | 2891  | (13) | 2895  | (13) | 73    | (14) | 333   | (14) |
|                           | 1117  | (14) | 351   | (16) | 428   | (16) | 1090  | (16) | 1093  | (16) | 1229  | (16) | 2030  | (16) | 669   | (18) |
|                           | 1237  | (18) | 158   | (23) | 175   | (23) | 143   | (25) | 269   | (27) | 101   | (28) | 107   | (28) | 758   | (28) |
|                           | 804   | (28) | 256   | (38) | 259   | (38) | 563   | (40) | 834   | (40) | 1076  | (40) | 441   | (41) |       |      |
| resTypeFound              | 158*  | (23) | 182-  | (23) | 211   | (23) |       |      |       |      |       |      |       |      |       |      |
| result                    | 2243* | ( 6) | 2272- | ( 6) | 2273  | ( 6) | 2277  | ( 6) | 843*  | (10) | 891-  | (10) | 902-  | (10) | 940*  | (10) |
|                           | 988-  | (10) | 999-  | (10) | 1519* | (10) | 1570- | (10) | 1584- | (10) | 1619* | (10) | 1670- | (10) | 1684- | (10) |
|                           | 278*  | (16) | 290   | (16) | 300-  | (16) | 305-  | (16) | 313   | (16) | 95*   | (26) | 109-  | (26) | 585*  | (26) |
|                           | 613-  | (26) | 615-  | (26) | 616   | (26) | 619-  | (26) | 621   | (26) | 648-  | (26) | 650-  | (26) | 651   | (26) |
|                           | 654-  | (26) | 660   | (26) | 617*  | (28) | 625-  | (28) | 633-  | (28) | 636   | (28) | 662-  | (28) | 664-  | (28) |
|                           | 668   | (28) | 685-  | (28) | 689   | (28) | 693-  | (28) | 696   | (28) | 159*  | (33) | 196-  | (33) | 202   | (33) |
|                           | 209-  | (33) | 212   | (33) | 216   | (33) | 276*  | (32) | 278   | (32) | 279   | (32) |       |      |       |      |
| RevealBottom              | 1615* | (13) | 1610* | (18) |       |      |       |      |       |      |       |      |       |      |       |      |
| RevealRect                | 3349  | ( 6) | 1871  | (10) | 1610* | (13) | 1845* | (13) | 1618  | (18) | 1625* | (18) | 1631  | (18) | 1641  | (18) |
|                           | 428*  | (20) | 1331  | (40) |       |      |       |      |       |      |       |      |       |      |       |      |
| RevealTop                 | 1613* | (13) | 2768  | (16) | 1639* | (18) |       |      |       |      |       |      |       |      |       |      |
| Revert                    | 1372* | (13) | 336   | (17) | 957*  | (17) |       |      |       |      |       |      |       |      |       |      |
| reverting                 | 1380* | (13) | 156*  | (17) | 161   | (17) | 172   | (17) |       |      |       |      |       |      |       |      |
| RevertSubview             | 1668* | (18) | 1676  | (18) |       |      |       |      |       |      |       |      |       |      |       |      |
| RevertView                | 1432* | (17) | 1437  | (17) |       |      |       |      |       |      |       |      |       |      |       |      |
| ReviveDeletions           | 442*  | (38) | 478*  | (38) | 213*  | (41) | 284   | (41) | 416*  | (41) | 419   | (41) |       |      |       |      |
| RGB                       | 342*  | (25) | 829*  | (26) | 832   | (26) | 833   | (26) | 834   | (26) |       |      |       |      |       |      |
| RGBBackColor              | 117   | ( 6) |       |      |       |      |       |      |       |      |       |      |       |      |       |      |
| RGBColor                  | 69    | ( 6) | 109   | ( 6) | 146   | ( 6) | 1255  | ( 6) | 2248  | ( 6) | 2249  | ( 6) | 2250  | ( 6) | 2251  | ( 6) |
|                           | 2295  | ( 6) | 2296  | ( 6) | 2297  | ( 6) | 2298  | ( 6) | 2807  | ( 6) | 113   | ( 9) | 2056  | (13) | 2116  | (13) |
|                           | 393   | (21) | 206   | (25) | 261   | (25) | 335   | (25) | 342   | (25) | 346   | (25) | 290   | (26) | 780   | (26) |

|                   |           |           |           |           |           |           |           |           |  |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
|                   | 829 (26)  | 842 (26)  | 892 (26)  | 82 (38)   |           |           |           |           |  |
| RGBForeColor      | 788 (26)  |           |           |           |           |           |           |           |  |
| RgnBBox           | 3083 (16) |           |           |           |           |           |           |           |  |
| rgnBBox           | 512 (10)  | 1152 (10) | 1333 (10) | 1427 (10) | 1775 (10) | 1862 (10) | 1863 (10) | 1864 (10) |  |
|                   | 2884 (10) | 1146 (16) | 3152 (16) | 165 (18)  | 1025 (18) | 83 (39)   | 84 (39)   | 88 (39)   |  |
|                   | 89 (39)   |           |           |           |           |           |           |           |  |
| rgnBBox           | 20 (18)   | 22 (18)   | 908 (18)  |           |           |           |           |           |  |
| RgnHandle         | 146 (9)   | 151 (9)   | 152 (9)   | 153 (9)   | 154 (9)   | 248 (9)   | 274 (9)   | 302 (9)   |  |
|                   | 374 (9)   | 643 (9)   | 644 (9)   | 46 (10)   | 143 (10)  | 488 (10)  | 563 (10)  | 1138 (10) |  |
|                   | 1984 (10) | 2783 (10) | 703 (13)  | 1630 (13) | 1633 (13) | 2639 (13) | 2762 (13) | 2796 (13) |  |
|                   | 2971 (13) | 814 (14)  | 816 (14)  | 1119 (16) | 424 (18)  | 425 (18)  | 842 (18)  | 875 (18)  |  |
|                   | 1263 (18) | 1555 (18) | 506 (19)  | 557 (19)  | 215 (38)  | 912 (40)  |           |           |  |
| right             | 2197 (6)  | 2214-(6)  | 2269 (6)  | 2346-(6)  | 2346 (6)  | 2381 (6)  | 2409 (6)  | 2410 (6)  |  |
|                   | 2640-(6)  | 3206-(6)  | 419-(10)  | 419 (10)  | 476-(10)  | 514-(10)  | 514 (10)  | 517 (10)  |  |
|                   | 520 (10)  | 703-(10)  | 703 (10)  | 731-(10)  | 733-(10)  | 736 (10)  | 799-(10)  | 801-(10)  |  |
|                   | 809 (10)  | 871-(10)  | 968-(10)  | 1430-(10) | 1430 (10) | 1435-(10) | 1435 (10) | 1479 (10) |  |
|                   | 1830-(10) | 1863 (10) | 1869 (10) | 1870 (10) | 2886-(10) | 2886 (10) | 2893-(10) | 2900-(10) |  |
|                   | 2956-(10) | 620 (14)  | 624 (14)  | 626 (14)  | 930-(14)  | 1032 (14) | 499 (18)  | 500 (18)  |  |
|                   | 507 (18)  | 511 (18)  | 513 (18)  | 1149-(18) | 479 (19)  | 567 (19)  | 567 (19)  | 568-(19)  |  |
|                   | 568 (19)  | 602 (19)  | 603 (19)  | 604 (19)  | 605 (19)  | 617 (19)  | 805 (19)  | 882-(19)  |  |
|                   | 882 (19)  | 968-(19)  | 976 (19)  | 1014 (19) | 226 (20)  | 264 (21)  | 783 (21)  | 785 (21)  |  |
|                   | 976 (21)  | 102 (23)  | 166*(25)  | 678 (26)  | 678 (26)  | 730 (26)  | 730 (26)  | 991 (26)  |  |
|                   | 1056 (26) | 369 (37)  | 543 (37)  | 557 (37)  | 1781-(37) | 1781 (37) | 283 (40)  | 381 (40)  |  |
|                   | 518 (40)  | 533-(40)  | 1068-(40) | 1068 (40) | 1068 (40) | 1221-(40) | 1221 (40) | 1269-(40) |  |
|                   | 1269 (40) | 1272 (40) | 1272 (40) | 1561-(40) | 1561 (40) | 50*(42)   |           |           |  |
| rightCD           | 176*(30)  |           |           |           |           |           |           |           |  |
| rightCd           | 183*(30)  |           |           |           |           |           |           |           |  |
| rightslop         | 2363*(6)  | 2381 (6)  |           |           |           |           |           |           |  |
| ROM85             | 78*(26)   |           |           |           |           |           |           |           |  |
| ROMMapInsert      | 1147*(26) | 1157 (26) |           |           |           |           |           |           |  |
| RoundUp           | 653 (18)  | 323*(25)  | 738*(26)  | 741-(26)  |           |           |           |           |  |
| Routine           | 13*(35)   | 22*(35)   | 32*(35)   | 117-(35)  | 161-(35)  | 196-(35)  |           |           |  |
| routine           | 319*(16)  | 73*(28)   |           |           |           |           |           |           |  |
| rowHeight         | 95*(9)    | 177*(9)   | 35*(10)   | 125 (10)  | 223 (10)  | 256-(10)  | 258-(10)  | 697*(10)  |  |
|                   | 745-(10)  | 770*(10)  | 784-(10)  |           |           |           |           |           |  |
| rowheight         | 750 (10)  | 790 (10)  |           |           |           |           |           |           |  |
| rowInset          | 97*(9)    | 181*(9)   | 436*(9)   | 529*(9)   | 39*(10)   | 57 (10)   | 58 (10)   | 60 (10)   |  |
|                   | 157 (10)  | 158 (10)  | 160 (10)  | 263-(10)  | 2208*(10) | 2218 (10) | 2219 (10) | 2448*(10) |  |
|                   | 2458 (10) |           |           |           |           |           |           |           |  |
| RowToVRect        | 270*(9)   | 414 (10)  | 973 (10)  | 1703 (10) | 1786*(10) |           |           |           |  |
| rPage             | 553 (37)  | 1751 (37) |           |           |           |           |           |           |  |
| rPaper            | 539 (37)  | 1754 (37) |           |           |           |           |           |           |  |
| rPlusL            | 355*(37)  | 369-(37)  | 371 (37)  |           |           |           |           |           |  |
| rsrcBase          | 103*(18)  | 108-(18)  | 114 (18)  | 120 (18)  |           |           |           |           |  |
| rsrcBytes         | 1046*(17) | 1127-(17) | 1128 (17) | 1129 (17) |           |           |           |           |  |
| rsrcExists        | 1180*(13) | 366*(17)  |           |           |           |           |           |           |  |
| rsrcForkBytes     | 1239*(13) | 352*(17)  | 357-(17)  | 357 (17)  |           |           |           |           |  |
| rsrcID            | 296*(5)   | 342*(5)   | 388*(5)   | 432*(5)   | 1453 (6)  | 1498-(6)  | 1676 (6)  | 1720-(6)  |  |
|                   | 1887 (6)  | 1921-(6)  | 2104 (6)  | 2107 (6)  | 2110 (6)  | 2155-(6)  | 253*(29)  | 258*(29)  |  |
|                   | 561*(30)  | 565 (30)  | 577*(30)  | 586 (30)  |           |           |           |           |  |
| rsrcId            | 921*(13)  | 1379*(13) | 1210*(16) | 1254 (16) | 1276 (16) | 1278 (16) | 156*(17)  | 166 (17)  |  |
| rsrcIndex         | 1379*(13) | 156*(17)  | 166 (17)  |           |           |           |           |           |  |
| rsrcPerm          | 1193*(13) | 721*(17)  | 725 (17)  | 414*(25)  | 580*(26)  | 653 (26)  |           |           |  |
| rsrcRefNum        | 865-(17)  | 866 (17)  | 657-(26)  |           |           |           |           |           |  |
| rsrcRefnum        | 1194*(13) | 629*(17)  | 639 (17)  | 660-(17)  | 705 (17)  | 711 (17)  | 722*(17)  | 726 (17)  |  |
|                   | 802*(17)  | 813 (17)  | 844-(17)  | 851 (17)  | 869 (17)  | 885 (17)  | 886-(17)  | 891 (17)  |  |
|                   | 1222*(17) | 1237 (17) | 1239 (17) | 1317*(17) | 1337 (17) | 1340 (17) | 380*(25)  | 415*(25)  |  |
|                   | 25*(26)   | 36 (26)   | 38 (26)   | 581*(26)  | 640-(26)  | 653-(26)  | 661-(26)  | 664-(26)  |  |
| rsrcSize          | 101*(18)  | 107-(18)  | 114 (18)  | 122 (18)  |           |           |           |           |  |
| rsrcType          | 1229*(16) |           |           |           |           |           |           |           |  |
| rType             | 269*(27)  | 101*(28)  | 113 (28)  | 115 (28)  | 121 (28)  |           |           |           |  |
| Run               | 770*(13)  | 2681*(16) |           |           |           |           |           |           |  |
| RunArray          | 82*(9)    | 83 (9)    |           |           |           |           |           |           |  |
| RunArrayChunk     | 77*(9)    | 82 (9)    |           |           |           |           |           |           |  |
| runs              | 1544 (40) |           |           |           |           |           |           |           |  |
| rview             | 1326*(6)  | 1337-(6)  | 1338 (6)  | 1339 (6)  |           |           |           |           |  |
| -S-               |           |           |           |           |           |           |           |           |  |
| s                 | 38*(8)    | 47 (8)    | 49 (8)    | 90*(8)    | 98 (8)    | 100 (8)   | 117*(8)   | 126 (8)   |  |
|                   | 128 (8)   | 590*(12)  | 594-(12)  | 596-(12)  | 597 (12)  | 601 (12)  | 603 (12)  | 608 (12)  |  |
|                   | 609 (12)  | 1955*(16) | 1998 (16) | 1999 (16) | 159*(17)  | 166 (17)  | 167 (17)  | 579*(17)  |  |
|                   | 596 (17)  | 597 (17)  | 736*(17)  | 738-(17)  | 739 (17)  | 369*(25)  | 430*(25)  | 442*(25)  |  |
|                   | 331*(26)  | 342-(26)  | 344-(26)  | 346 (26)  | 348 (26)  | 712*(26)  | 717 (26)  | 719 (26)  |  |
|                   | 719 (26)  | 910*(26)  | 913 (26)  | 1127*(26) | 1133 (26) | 1134 (26) | 1135-(26) | 1135 (26) |  |
|                   | 156*(28)  | 172 (28)  | 177 (28)  | 303*(28)  | 311 (28)  | 312-(28)  | 312 (28)  | 313 (28)  |  |
|                   | 331*(28)  | 365 (28)  | 366 (28)  | 367 (28)  | 367 (28)  | 563*(30)  | 565 (30)  | 566 (30)  |  |
|                   | 68*(33)   | 76 (33)   | 79 (33)   | 165*(33)  | 182-(33)  | 184-(33)  | 186 (33)  | 155*(32)  |  |
|                   | 162 (32)  | 163 (32)  | 174 (32)  |           |           |           |           |           |  |
| SameCell          | 356*(9)   | 1841*(10) | 1843-(10) | 2943 (10) |           |           |           |           |  |
| Sane              | 20*(5)    |           |           |           |           |           |           |           |  |
| Save              | 1292*(13) | 205 (17)  | 294 (17)  | 1036*(17) |           |           |           |           |  |
| SaveAgain         | 1301*(13) | 940 (17)  | 1206*(17) |           |           |           |           |           |  |
| saveCmd           | 2247*(16) | 2279-(16) | 2281 (16) | 2295 (16) | 2309 (16) |           |           |           |  |
| savedChar         | 711*(41)  | 730-(41)  | 733 (41)  | 777 (41)  |           |           |           |           |  |
| savedClip         | 875*(18)  | 891 (18)  | 922-(18)  | 925 (18)  | 927 (18)  |           |           |           |  |
| savedLongOffset   | 877*(18)  | 892 (18)  | 924-(18)  |           |           |           |           |           |  |
| SavedOn           | 1312*(13) | 1196 (17) | 1219*(17) |           |           |           |           |           |  |
| savedOrigin       | 876*(18)  | 890 (18)  | 890 (18)  | 923-(18)  |           |           |           |           |  |
| savedOwner        | 693*(21)  | 699-(21)  | 707 (21)  |           |           |           |           |           |  |
| savedPenState     | 399*(18)  | 479 (18)  | 515 (18)  |           |           |           |           |           |  |
| savedPerm         | 1241*(18) | 1247-(18) | 1249-(18) | 1249 (18) | 159*(23)  | 210-(23)  | 212-(23)  | 212 (23)  |  |
|                   | 1083*(40) | 1106-(40) | 1116-(40) | 1119-(40) | 1119 (40) | 1146-(40) | 1153-(40) | 1153 (40) |  |
|                   | 436*(41)  | 449-(41)  | 449 (41)  | 458-(41)  | 500-(41)  | 508-(41)  | 508 (41)  | 522*(41)  |  |
|                   | 714*(41)  | 818*(41)  | 826-(41)  | 826 (41)  | 837-(41)  | 852-(41)  | 852 (41)  |           |  |
| savedPort         | 629*(19)  | 634 (19)  | 639 (19)  | 57*(37)   | 66 (37)   | 103 (37)  | 302*(40)  | 320 (40)  |  |
|                   | 324 (40)  | 415*(40)  | 444 (40)  | 448 (40)  | 984*(40)  | 990-(40)  | 1025 (40) | 1364*(40) |  |
|                   | 1405 (40) | 1409 (40) |           |           |           |           |           |           |  |
| savedPrintHandler | 64*(37)   | 95-(37)   | 99 (37)   |           |           |           |           |           |  |
| savedProcs        | 426*(30)  | 447-(30)  | 488 (30)  |           |           |           |           |           |  |

|                         |            |            |            |            |            |            |            |            |
|-------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| savedSize               | 123* (41)  | 130* (41)  | 148 (41)   | 216* (41)  | 224* (41)  | 237 (41)   | 705* (41)  | 718* (41)  |
|                         | 720 (41)   | 721* (41)  | 721 (41)   | 722 (41)   | 725 (41)   | 728 (41)   | 733 (41)   | 775 (41)   |
|                         | 777 (41)   | 790* (41)  | 792* (41)  | 794 (41)   | 795* (41)  | 795 (41)   | 796 (41)   | 798 (41)   |
|                         | 799 (41)   |            |            |            |            |            |            |            |
| saveEnd                 | 1363* (40) | 1376* (40) | 1380 (40)  |            |            |            |            |            |
| SaveInPlace             | 1316* (13) | 1173 (17)  | 1247* (17) |            |            |            |            |            |
| SavePrArray             | 421* (30)  | 426 (30)   |            |            |            |            |            |            |
| saveStart               | 1362* (40) | 1375* (40) | 1380 (40)  |            |            |            |            |            |
| SaveViaTemp             | 1332* (13) | 1133 (17)  | 1287* (17) |            |            |            |            |            |
| savingDoc               | 1302* (13) | 1207* (17) | 1210 (17)  |            |            |            |            |            |
| SBarActionProc          | 12* (21)   | 1059 (21)  |            |            |            |            |            |            |
| sBarDelta               | 1345* (21) | 1353* (21) | 1353 (21)  | 1362* (21) | 1371 (21)  | 1372 (21)  | 1407* (21) | 1415* (21) |
|                         | 1415 (21)  | 1419* (21) | 1422 (21)  |            |            |            |            |            |
| sBarLocation            | 209* (20)  | 225 (20)   | 229 (20)   | 237 (20)   | 241 (20)   |            |            |            |
| sBarOffsets             | 1753* (13) | 905* (14)  | 924* (14)  | 930 (14)   | 936 (14)   | 942 (14)   | 66 (20)    | 101 (20)   |
| sBarSize                | 210* (20)  | 226 (20)   | 229 (20)   | 238 (20)   | 241 (20)   |            |            |            |
| sBarValue               | 1837* (13) | 528* (20)  | 535 (20)   | 536 (20)   | 539 (20)   | 544 (20)   |            |            |
| sBarWasVisible          | 401* (20)  | 409* (20)  | 413* (20)  | 420 (20)   |            |            |            |            |
| sbPtr                   | 994* (21)  | 999* (21)  | 1002 (21)  |            |            |            |            |            |
| ScanHandles             | 276* (27)  | 948* (28)  | 1094 (28)  |            |            |            |            |            |
| ScanList                | 956* (28)  | 976 (28)   | 978 (28)   | 979 (28)   | 981 (28)   |            |            |            |
| scPtr                   | 83* (20)   | 88* (20)   | 91 (20)    |            |            |            |            |            |
| scrapCount              | 382 (16)   | 382 (16)   | 29 (23)    | 51 (23)    | 191 (23)   | 266 (23)   | 271 (23)   |            |
| scrapState              | 392 (16)   |            |            |            |            |            |            |            |
| ScrapStuff              | 2721 (13)  | 2738 (13)  |            |            |            |            |            |            |
| screenBits              | 524 (14)   | 618 (14)   | 313 (19)   | 314 (19)   | 357 (19)   | 358 (19)   |            |            |
| screenSize              | 349* (19)  | 357* (19)  | 358 (19)   | 363 (19)   | 365 (19)   |            |            |            |
| Script                  | 21* (5)    | 47* (13)   | 46* (25)   | 27* (38)   |            |            |            |            |
| scrollBarProc           | 946 (21)   | 971 (21)   |            |            |            |            |            |            |
| ScrollBarTemplate       | 2214 (13)  | 2215* (13) | 981 (21)   | 1000 (21)  |            |            |            |            |
| ScrollBarTemplatePtr    | 2214* (13) | 972 (21)   | 994 (21)   | 999 (21)   |            |            |            |            |
| ScrollBy                | 1839* (13) | 450 (20)   | 457* (20)  | 595 (20)   | 36 (22)    | 217 (39)   |            |            |
| ScrollDraw              | 1831* (13) | 284 (20)   | 479* (20)  |            |            |            |            |            |
| scroller                | 3103* (16) | 3131 (16)  | 3133 (16)  | 3138 (16)  | 3163 (16)  | 3163 (16)  | 3164 (16)  | 3180* (16) |
|                         | 3299 (16)  | 3303 (16)  | 186* (39)  | 193* (39)  | 194* (39)  | 195 (39)   | 195 (39)   | 198 (39)   |
|                         | 217 (39)   |            |            |            |            |            |            |            |
| ScrollerTemplate        | 1743 (13)  | 1744* (13) | 71 (20)    | 89 (20)    |            |            |            |            |
| ScrollerTemplatePtr     | 1743* (13) | 57 (20)    | 83 (20)    | 88 (20)    |            |            |            |            |
| scrollLimit             | 1819* (13) | 18* (20)   | 32 (20)    | 33 (20)    | 34 (20)    | 49* (20)   | 60 (20)    | 61 (20)    |
|                         | 62 (20)    | 602* (20)  | 610 (20)   | 614 (20)   |            |            |            |            |
| scrollLimits            | 555* (18)  | 560* (18)  | 561 (18)   | 562 (18)   |            |            |            |            |
| ScrollRect              | 509 (20)   |            |            |            |            |            |            |            |
| ScrollRelative          | 1837* (13) | 528* (20)  | 544* (20)  | 1353 (21)  |            |            |            |            |
| ScrollSelectionIntoView | 685* (5)   | 2648* (6)  | 359* (9)   | 1852* (10) | 297* (38)  | 711 (40)   | 868 (40)   | 1290* (40) |
|                         | 1590 (40)  |            |            |            |            |            |            |            |
| ScrollStep              | 1835* (13) | 551* (20)  | 585* (20)  | 1415 (21)  |            |            |            |            |
| ScrollTo                | 1841* (13) | 592* (20)  | 628 (20)   |            |            |            |            |            |
| ScrollToThumb           | 1350* (21) | 1369 (21)  |            |            |            |            |            |            |
| scrollUnit              | 554* (20)  | 559* (20)  | 564 (20)   | 570 (20)   |            |            |            |            |
| ScrollView              | 1412* (21) | 1420 (21)  |            |            |            |            |            |            |
| scrpAscent              | 765* (41)  |            |            |            |            |            |            |            |
| scrpFont                | 130 (39)   | 147 (39)   | 741 (41)   | 766 (41)   |            |            |            |            |
| scrpHeight              | 764* (41)  |            |            |            |            |            |            |            |
| scrpNStyles             | 124 (39)   | 141 (39)   | 757 (41)   | 760* (41)  | 761 (41)   | 921 (41)   |            |            |
| scrpStartChar           | 129 (39)   | 146 (39)   | 755 (41)   | 756 (41)   | 772* (41)  | 772 (41)   | 919 (41)   | 922* (41)  |
|                         | 923 (41)   |            |            |            |            |            |            |            |
| ScrpSElement            | 1468 (40)  | 749 (41)   | 751 (41)   | 756 (41)   |            |            |            |            |
| scrpStyleTab            | 127 (39)   | 144 (39)   | 741 (41)   | 754 (41)   | 762 (41)   | 771 (41)   | 919 (41)   | 922 (41)   |
|                         | 923 (41)   |            |            |            |            |            |            |            |
| Search                  | 156 (2)    | 287 (2)    | 338 (2)    | 283* (11)  | 726* (12)  | 745* (12)  | 746 (12)   | 759* (12)  |
|                         | 760 (12)   | 769* (12)  |            |            |            |            |            |            |
| SearchTable             | 720* (14)  | 736* (14)  | 759* (14)  | 771 (14)   | 774 (14)   |            |            |            |
| second                  | 240* (25)  | 166* (26)  |            |            |            |            |            |            |
| SectRect                | 2306 (6)   | 2347 (6)   | 2379 (6)   | 908 (18)   | 961 (37)   |            |            |            |
| SectRgn                 | 2017 (10)  | 3059 (16)  | 164 (18)   | 588 (18)   |            |            |            |            |
| SectVRect               | 1029 (18)  | 58* (42)   |            |            |            |            |            |            |
| seg                     | 149* (8)   | 154* (28)  | 177* (28)  | 178 (28)   | 179 (28)   | 411* (28)  | 451* (28)  | 452 (28)   |
|                         | 453 (28)   | 573* (28)  | 803* (28)  | 819* (28)  | 820 (28)   | 822 (28)   | 902* (28)  | 913* (28)  |
|                         | 915 (28)   | 921 (28)   | 923 (28)   | 924 (28)   | 1114* (28) | 1134* (28) | 1135 (28)  | 1136 (28)  |
|                         | 1137 (28)  |            |            |            |            |            |            |            |
| SegHeader               | 894 (28)   | 895* (28)  |            |            |            |            |            |            |
| segName                 | 759* (28)  | 776 (28)   | 778 (28)   |            |            |            |            |            |
| segNum                  | 228* (27)  | 240* (27)  | 248* (27)  | 572* (28)  | 746* (28)  | 767 (28)   | 769 (28)   | 776 (28)   |
|                         | 777 (28)   | 891* (28)  | 908 (28)   | 913 (28)   | 1013* (28) | 1015 (28)  | 1018 (28)  | 1019 (28)  |
|                         | 1024 (28)  | 1024 (28)  | 1025 (28)  |            |            |            |            |            |
| segnum                  | 224* (27)  | 597* (28)  | 604 (28)   | 1020 (28)  |            |            |            |            |
| segNumber               | 800* (28)  | 822 (28)   | 823 (28)   |            |            |            |            |            |
| segRsrc                 | 357* (27)  | 148* (28)  | 158 (28)   | 159 (28)   | 164 (28)   | 186 (28)   |            |            |
| segStart                | 574* (28)  |            |            |            |            |            |            |            |
| selChars                | 15* (41)   | 35* (41)   | 55 (41)    | 58 (41)    | 59 (41)    |            |            |            |
| Select                  | 2004* (13) | 748 (16)   | 745 (17)   | 699 (19)   | 773* (19)  |            |            |            |
| select                  | 366* (9)   | 374* (9)   | 381* (9)   | 603* (9)   | 606* (9)   | 1933* (10) | 1942 (10)  | 1985* (10) |
|                         | 2004 (10)  | 2010 (10)  | 2022 (10)  | 2024 (10)  | 2045* (10) | 2049 (10)  | 2710* (10) | 2712 (10)  |
|                         | 2721* (10) | 2728 (10)  |            |            |            |            |            |            |
| SelectCell              | 366* (9)   | 603* (9)   | 1933* (10) | 2709* (10) | 2728 (10)  | 3040 (10)  |            |            |
| selectChars             | 98* (5)    | 118* (5)   | 616* (5)   | 679* (5)   | 509* (6)   | 522 (6)    | 530 (6)    | 698* (6)   |
|                         | 706 (6)    | 2576* (6)  | 2600 (6)   | 3330* (6)  | 3338 (6)   |            |            |            |
| SelectEditText          | 118* (5)   | 698* (6)   |            |            |            |            |            |            |
| selectionRect           | 1857* (10) | 1866 (10)  | 1871 (10)  | 270* (38)  | 391* (40)  | 435 (40)   | 437 (40)   | 437 (40)   |
|                         | 438 (40)   | 449 (40)   | 449 (40)   | 451 (40)   | 451 (40)   | 469 (40)   | 516 (40)   | 518 (40)   |
|                         | 521 (40)   | 524 (40)   |            |            |            |            |            |            |
| SelectItem              | 606* (9)   | 2712 (10)  | 2721* (10) |            |            |            |            |            |
| SelectWindow            | 2737 (16)  |            |            |            |            |            |            |            |
| SelectWmgrWindow        | 916* (13)  | 758 (16)   | 1586 (16)  | 2734* (16) | 776 (19)   | 74 (37)    |            |            |
| selEnd                  | 612* (5)   | 3267* (6)  | 3272 (6)   | 3274 (6)   | 168* (39)  | 423 (40)   | 552 (40)   | 719 (40)   |
|                         | 729 (40)   | 853 (40)   | 958 (40)   | 1135 (40)  | 1376 (40)  | 34 (41)    | 35 (41)    | 77 (41)    |
|                         | 833 (41)   |            |            |            |            |            |            |            |
| SELF                    | 498 (6)    | 500 (6)    | 527 (6)    | 563 (6)    | 619 (6)    | 1236 (6)   | 2280 (6)   | 2585 (6)   |
|                         | 2586 (6)   | 2607 (6)   | 3068 (6)   | 3069 (6)   | 3072 (6)   | 3338 (6)   | 3342 (6)   | 642 (10)   |

|                     |            |            |            |            |            |            |           |           |
|---------------------|------------|------------|------------|------------|------------|------------|-----------|-----------|
|                     | 651 (10)   | 660 (10)   | 669 (10)   | 3048 (10)  | 62 (12)    | 157 (12)   | 207 (12)  | 207 (12)  |
|                     | 228 (12)   | 281 (12)   | 326 (12)   | 450 (12)   | 455 (12)   | 461 (12)   | 523 (12)  | 542 (12)  |
|                     | 542 (12)   | 559 (12)   | 576 (12)   | 26 (15)    | 76 (16)    | 79 (16)    | 85 (16)   | 1081 (16) |
|                     | 104 (17)   | 195 (17)   | 209 (17)   | 940 (17)   | 988 (17)   | 1107 (17)  | 1210 (17) | 208 (18)  |
|                     | 210 (18)   | 293 (18)   | 295 (18)   | 356 (18)   | 358 (18)   | 367 (18)   | 893 (18)  | 967 (18)  |
|                     | 1013 (18)  | 1016 (18)  | 1019 (18)  | 1049 (18)  | 1288 (18)  | 1417 (18)  | 1451 (18) | 1541 (18) |
|                     | 1601 (18)  | 1819 (18)  | 44 (19)    | 54 (19)    | 121 (19)   | 124 (19)   | 130 (19)  | 237 (19)  |
|                     | 239 (19)   | 272 (19)   | 381 (19)   | 509 (19)   | 523 (19)   | 676 (19)   | 728 (19)  | 729 (19)  |
|                     | 737 (19)   | 896 (19)   | 141 (20)   | 230 (20)   | 242 (20)   | 280 (20)   | 342 (20)  | 85 (21)   |
|                     | 309 (21)   | 483 (21)   | 643 (21)   | 682 (21)   | 1232 (21)  | 1304 (21)  | 101 (22)  | 285 (23)  |
|                     | 287 (23)   | 126 (24)   | 196 (33)   | 240 (33)   | 253 (33)   | 278 (32)   | 307 (32)  | 292 (37)  |
|                     | 302 (37)   | 324 (37)   | 330 (37)   | 1158 (37)  | 1345 (37)  | 769 (40)   | 776 (40)  | 783 (40)  |
|                     | 803 (40)   | 819 (40)   | 888 (40)   | 917 (40)   | 1188 (40)  | 270 (41)   | 290 (41)  | 307 (41)  |
|                     | 406 (41)   | 574 (41)   | 591 (41)   | 667 (41)   | 869 (41)   | 885 (41)   | 900 (41)  | 927 (41)  |
| sellLine            | 404* (40)  | 459* (40)  | 460 (40)   | 461 (40)   | 462* (40)  | 462 (40)   | 463* (40) | 463 (40)  |
|                     | 466 (40)   | 467* (40)  | 467 (40)   | 469 (40)   | 470 (40)   | 471 (40)   | 472 (40)  |           |
| selLoc              | 267* (40)  | 1294* (40) | 1311 (40)  |            |            |            |           |           |
| selRect             | 850* (10)  | 868 (10)   | 869 (10)   | 870 (10)   | 871 (10)   | 872 (10)   | 947* (10) | 965 (10)  |
|                     | 966 (10)   | 967 (10)   | 968 (10)   | 969 (10)   |            |            |           |           |
| selStart            | 612* ( 5)  | 3267* ( 6) | 3272 ( 6)  | 3274 ( 6)  | 167* (39)  | 422 (40)   | 551 (40)  | 719 (40)  |
|                     | 729 (40)   | 853 (40)   | 958 (40)   | 1134 (40)  | 1375 (40)  | 33 (41)    | 35 (41)   | 77 (41)   |
|                     | 485 (41)   | 649 (41)   | 832 (41)   |            |            |            |           |           |
| SetApplLimit        | 1049 (28)  |            |            |            |            |            |           |           |
| SetBadColors        | 157* ( 6)  | 189 ( 6)   | 226 ( 6)   | 229 ( 6)   |            |            |           |           |
| SetCCursor          | 215 ( 4)   |            |            |            |            |            |           |           |
| SetChooserAlert     | 112 (14)   | 546 (14)   |            |            |            |            |           |           |
| SetClickLoop        | 54 (40)    | 110 (40)   |            |            |            |            |           |           |
| SetClip             | 1196 (14)  | 590 (18)   | 891 (18)   | 490 (19)   | 1003 (40)  | 1019 (40)  |           |           |
| SetClipView         | 1052* (13) | 483 (16)   | 2257 (16)  | 2746* (16) | 2986 (16)  |            |           |           |
| SetCMacAppCursor    | 62* ( 4)   | 285* ( 4)  |            |            |            |            |           |           |
| SetCMacappCursor    | 146 ( 4)   |            |            |            |            |            |           |           |
| SetCmdIcon          | 243* (29)  | 517* (30)  |            |            |            |            |           |           |
| SetCmdName          | 2828 (16)  | 248* (29)  | 539* (30)  | 566 (30)   |            |            |           |           |
| SetColWidth         | 362* ( 9)  | 1882* (10) | 2223 (10)  | 2256 (10)  | 2751 (10)  |            |           |           |
| SetTitle            | 841 (21)   |            |            |            |            |            |           |           |
| SetCtlMax           | 805 (21)   | 878 (21)   |            |            |            |            |           |           |
| SetCtlMin           | 823 (21)   | 877 (21)   |            |            |            |            |           |           |
| SetCtlValue         | 859 (21)   | 879 (21)   |            |            |            |            |           |           |
| SetCurrentItem      | 484* ( 5)  | 2279 ( 6)  | 2460* ( 6) | 2516 ( 6)  |            |            |           |           |
| SetCursor           | 217 ( 4)   | 803 (14)   | 1594 (16)  | 3090 (16)  | 282 (19)   | 920 (40)   |           |           |
| SetDefaultPrintInfo | 383* (36)  | 291 (37)   | 1790* (37) |            |            |            |           |           |
| SetEltType          | 102 ( 2)   | 240* (11)  | 586* (12)  | 654 (14)   | 92 (16)    | 85 (17)    | 90 (17)   | 346 (18)  |
|                     | 1227 (21)  | 1261 (21)  |            |            |            |            |           |           |
| SetEmptyRgn         | 508 (10)   | 531 (10)   | 1937 (10)  | 1954 (10)  | 133 (14)   | 805 (14)   | 2794 (16) | 3038 (16) |
| SetEmptySelection   | 370* ( 9)  | 1952* (10) |            |            |            |            |           |           |
| SetFocus            | 1191* (14) | 3167 (16)  | 912 (21)   | 1370 (21)  | 1421 (21)  |            |           |           |
| SetFPos             | 861 (17)   |            |            |            |            |            |           |           |
| SetFractEnable      | 423 (14)   |            |            |            |            |            |           |           |
| SetGrowZone         | 436 (28)   |            |            |            |            |            |           |           |
| SetHandleBits       | 1550 ( 6)  | 1770 ( 6)  | 1969 ( 6)  | 332* (25)  | 764* (26)  |            |           |           |
| SetHandleSize       | 228 (12)   | 576 (12)   | 2938 (16)  | 68 (18)    | 122 (18)   | 459 (28)   | 1117 (40) | 113 (41)  |
|                     | 195 (41)   | 501 (41)   | 684 (41)   | 748 (41)   | 775 (41)   | 796 (41)   | 801 (41)  |           |
| SetHilite           | 661* (21)  | 669 (21)   | 762* (21)  | 769 (21)   |            |            |           |           |
| SetHLPenState       | 2982* (13) | 1205* (14) |            |            |            |            |           |           |
| SetIcon             | 327* ( 5)  | 1577* ( 6) |            |            |            |            |           |           |
| SetIfBKColor        | 109* ( 6)  | 2264 ( 6)  | 2274 ( 6)  | 2276 ( 6)  | 2283 ( 6)  | 2314 ( 6)  | 2319 ( 6) | 2323 ( 6) |
| SetIfColor          | 1292 ( 6)  | 2264 ( 6)  | 2274 ( 6)  | 2276 ( 6)  | 2283 ( 6)  | 2314 ( 6)  | 2319 ( 6) | 2323 ( 6) |
|                     | 2404 ( 6)  | 2824 ( 6)  | 335* (25)  | 780* (26)  | 822 (26)   | 1418 (40)  |           |           |
| SetIndCmdName       | 253* (29)  | 561* (30)  | 586 (30)   |            |            |            |           |           |
| SetInitHandler      | 221* ( 7)  | 284* ( 8)  | 451 (14)   | 2690 (16)  |            |            |           |           |
| SetInset            | 2114* (13) | 438* (21)  |            |            |            |            |           |           |
| SetItem             | 551 (30)   |            |            |            |            |            |           |           |
| SetItemHeight       | 611* ( 9)  | 2737* (10) |            |            |            |            |           |           |
| SetItemIcon         | 529 (30)   |            |            |            |            |            |           |           |
| SetItemMark         | 2467 ( 6)  | 2469 ( 6)  |            |            |            |            |           |           |
| SetItemStyle        | 608 (30)   |            |            |            |            |            |           |           |
| SetItemWidth        | 616* ( 9)  | 2749* (10) |            |            |            |            |           |           |
| SetIText            | 1404 (37)  |            |            |            |            |            |           |           |
| SetJustification    | 541* ( 5)  | 610* ( 5)  | 2582 ( 6)  | 2866* ( 6) | 3256* ( 6) | 3259 ( 6)  | 3260 ( 6) | 301* (38) |
|                     | 1234 (40)  | 1348* (40) |            |            |            |            |           |           |
| SetKeyScript        | 326* (25)  | 748* (26)  | 1186 (40)  |            |            |            |           |           |
| SetLabel            | 281* ( 5)  | 1145 ( 6)  | 1166 ( 6)  | 1348* ( 6) |            |            |           |           |
| SetLongMax          | 2257* (13) | 619 (20)   | 1108* (21) | 1160 (21)  |            |            |           |           |
| SetLongMin          | 2259* (13) | 1128* (21) | 1161 (21)  |            |            |            |           |           |
| SetLongVal          | 2261* (13) | 1040 (21)  | 1142* (21) | 1162 (21)  | 1372 (21)  |            |           |           |
| SetLongValues       | 2263* (13) | 948 (21)   | 973 (21)   | 1157* (21) |            |            |           |           |
| SetMacAppCursor     | 61* ( 4)   | 257 ( 4)   | 312* ( 4)  |            |            |            |           |           |
| SetMacappCursor     | 143 ( 4)   |            |            |            |            |            |           |           |
| SetMargins          | 390* (36)  | 1636 (37)  | 1831* (37) |            |            |            |           |           |
| SetMax              | 2192* (13) | 798* (21)  | 1120 (21)  |            |            |            |           |           |
| SetMCEntries        | 30 (16)    | 43 (16)    | 378 (30)   |            |            |            |           |           |
| SetMenuBar          | 107 (16)   |            |            |            |            |            |           |           |
| SetMenuState        | 849 (16)   | 2900 (16)  | 2904 (16)  | 257* (29)  | 576* (30)  |            |           |           |
| SetMin              | 2194* (13) | 816* (21)  | 1134 (21)  |            |            |            |           |           |
| SetOneStyle         | 2589 ( 6)  | 306* (38)  | 1360* (40) | 544 (41)   |            |            |           |           |
| SetOrigin           | 1195 (14)  | 890 (18)   | 1045 (18)  | 520 (19)   | 919 (37)   | 953 (37)   |           |           |
| SetPage             | 345* (36)  | 1593 (37)  | 1844* (37) |            |            |            |           |           |
| SetPageInterior     | 2415* (13) | 211* (24)  | 348* (36)  | 1643 (37)  | 1859 (37)  | 1881* (37) |           |           |
| SetPageOffset       | 2418* (13) | 834 (18)   | 220* (24)  | 353* (36)  | 1909* (37) |            |           |           |
| SetPattern          | 373* ( 5)  | 1797* ( 6) |            |            |            |            |           |           |
| SetPen              | 488* ( 9)  | 2338 (10)  | 2356 (10)  | 2372* (10) |            |            |           |           |
| SetPenState         | 515 (18)   | 752 (37)   |            |            |            |            |           |           |
| SetPicture          | 418* ( 5)  | 1989* ( 6) |            |            |            |            |           |           |
| SetPopup            | 486* ( 5)  | 2054 ( 6)  | 2110 ( 6)  | 2484* ( 6) |            |            |           |           |
| SetPort             | 1194 (14)  | 3151 (16)  | 39 (19)    | 475 (19)   | 519 (19)   | 635 (19)   | 639 (19)  | 103 (37)  |
|                     | 1162 (37)  | 321 (40)   | 324 (40)   | 445 (40)   | 448 (40)   | 1217 (40)  | 1229 (40) | 1406 (40) |
|                     | 1409 (40)  |            |            |            |            |            |           |           |
| SetPortTextStyle    | 1263 ( 6)  | 2817 ( 6)  | 2378 (10)  | 341 (21)   | 89 (23)    | 242 (23)   | 338* (25) | 816* (26) |
|                     | 748 (37)   | 322 (40)   | 446 (40)   | 1218 (40)  | 1407 (40)  |            |           |           |

|                     |           |           |           |           |           |           |           |           |  |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| SetPrintExtent      | 386*(36)  | 742 (37)  | 1546 (37) | 1633 (37) | 1772*(37) |           |           |           |  |
| SetPt               | 2269 ( 6) | 1863 (10) | 277 (37)  | 550 (37)  | 2037 (37) | 1319 (40) |           |           |  |
| SetRect             | 1285 ( 6) | 620 (14)  | 624 (14)  | 626 (14)  | 110 (19)  | 829 (19)  | 264 (21)  | 541 (21)  |  |
|                     | 783 (21)  | 182 (37)  | 555 (37)  | 1751 (37) | 1754 (37) | 369 (41)  |           |           |  |
| SetRectRgn          | 1939 (10) | 2047 (10) | 3074 (16) |           |           |           |           |           |  |
| SetReserveSize      | 2202 (16) | 2213 (16) | 2232 (16) | 337*(27)  | 374 (28)  | 734 (28)  | 991*(28)  |           |  |
| SetResidentSegment  | 358 (14)  | 369 (14)  | 2698 (16) | 2703 (16) | 248*(27)  | 823 (28)  | 1013*(28) |           |  |
| SetResizerLimits    | 2022*(13) | 64 (19)   | 139 (19)  | 866*(19)  |           |           |           |           |  |
| SetResLoad          | 110 (14)  | 1153 (26) | 111 (28)  | 128 (28)  | 162 (28)  | 184 (28)  | 448 (28)  | 457 (28)  |  |
|                     | 603 (28)  | 605 (28)  |           |           |           |           |           |           |  |
| SetRGBColor         | 414 (14)  | 415 (14)  | 342*(25)  | 829*(26)  |           |           |           |           |  |
| SetRowHeight        | 363*( 9)  | 1908*(10) | 2252 (10) | 2740 (10) |           |           |           |           |  |
| SetScrollBar        | 1321*(21) | 1334 (21) |           |           |           |           |           |           |  |
| SetScrollLimits     | 1819*(13) | 562 (18)  | 34 (20)   | 62 (20)   | 417 (20)  | 602*(20)  | 654 (20)  |           |  |
| SetScrollParameters | 1821*(13) | 30 (20)   | 64 (20)   | 637*(20)  | 254 (40)  |           |           |           |  |
| SetSelect           | 2601 ( 6) | 2603 ( 6) | 3274 ( 6) | 605*(38)  | 163*(39)  | 554 (40)  | 556 (40)  | 1137 (40) |  |
|                     | 1139 (40) | 1378 (40) | 1380 (40) | 1607 (40) |           |           |           |           |  |
| SetSelection        | 612*( 5)  | 531 ( 6)  | 3267*( 6) | 374*( 9)  | 1942 (10) | 1955 (10) | 1984*(10) | 2049 (10) |  |
|                     | 3042 (10) |           |           |           |           |           |           |           |  |
| SetSelRect          | 380*( 9)  | 2044*(10) |           |           |           |           |           |           |  |
| SetSingleSelection  | 385*( 9)  | 2058*(10) |           |           |           |           |           |           |  |
| SetStackSpace       | 361*(27)  | 503 (28)  | 1036*(28) |           |           |           |           |           |  |
| SetState            | 190*( 5)  | 230*( 5)  | 867 ( 6)  | 956*( 6)  | 995 ( 6)  | 1085*( 6) | 1231 ( 6) |           |  |
| SetString           | 220 ( 2)  |           |           |           |           |           |           |           |  |
| SetStyle            | 265*(29)  | 596*(30)  |           |           |           |           |           |           |  |
| SetStyleHandle      | 1501 (40) |           |           |           |           |           |           |           |  |
| SetStyleScrap       | 40*(38)   | 391 (41)  | 556 (41)  |           |           |           |           |           |  |
| SetTarget           | 527 ( 6)  | 3342 ( 6) | 812*(13)  | 2024*(13) | 2787*(16) | 273 (19)  | 279 (19)  | 892*(19)  |  |
|                     | 897 (19)  |           |           |           |           |           |           |           |  |
| SetText             | 543*( 5)  | 614*( 5)  | 2598 ( 6) | 2695 ( 6) | 2718 ( 6) | 2877*( 6) | 3281*( 6) | 3293 ( 6) |  |
|                     | 3304 ( 6) | 3369 ( 6) | 3405 ( 6) | 3426 ( 6) | 3501 ( 6) | 2196*(13) | 834*(21)  | 312*(38)  |  |
|                     | 1437*(40) |           |           |           |           |           |           |           |  |
| SetTextStyle        | 2244 (10) | 687 (14)  | 148 (21)  | 345*(25)  | 841*(26)  | 747 (37)  | 98 (40)   | 365 (41)  |  |
| SetTitle            | 1398*(13) | 2158 (16) | 1229 (17) | 1380*(17) |           |           |           |           |  |
| SetTitleForDoc      | 2006*(13) | 1387 (17) | 732 (19)  | 905*(19)  |           |           |           |           |  |
| SetUndoText         | 951*(13)  | 2801*(16) | 2887 (16) |           |           |           |           |           |  |
| SetupA5             | 273 ( 4)  | 292 ( 4)  | 319 ( 4)  | 91 (14)   | 623 (28)  | 763 (28)  |           |           |  |
| SetupFocus          | 3122*(16) | 3165 (16) | 3246 (16) | 3319 (16) |           |           |           |           |  |
| SetupFont           | 485*( 9)  | 2215 (10) | 2248 (10) | 2351*(10) |           |           |           |           |  |
| SetupForFinder      | 2395*(13) | 2459 (16) | 180*(24)  | 185-(24)  | 410*(36)  | 1923*(37) | 1960-(37) |           |  |
| SetupPrintOne       | 416*(36)  | 671 (37)  | 1968*(37) | 1983-(37) |           |           |           |           |  |
| SetupTheMenus       | 957*(13)  | 675 (16)  | 1600 (16) | 1786 (16) | 2398 (16) | 2709 (16) | 2839*(16) |           |  |
| SetVal              | 958 ( 6)  | 967 ( 6)  | 977 ( 6)  | 1087 ( 6) | 1096 ( 6) | 1106 ( 6) | 2198*(13) | 852*(21)  |  |
|                     | 1149 (21) |           |           |           |           |           |           |           |  |
| SetValue            | 660*( 5)  | 3493*( 6) |           |           |           |           |           |           |  |
| SetValues           | 2200*(13) | 870*(21)  |           |           |           |           |           |           |  |
| SetVol              | 674 (17)  | 680 (17)  | 638 (26)  | 642 (26)  |           |           |           |           |  |
| SetVpt              | 605 (14)  | 862 (14)  | 3291 (16) | 1570 (18) | 225 (20)  | 226 (20)  | 237 (20)  | 238 (20)  |  |
|                     | 470 (20)  | 21 (23)   | 22 (23)   | 283 (40)  | 368 (41)  | 39*(42)   |           |           |  |
|                     | 410 (10)  | 449 (10)  | 1803 (10) | 606 (14)  | 50*(42)   |           |           |           |  |
| SetVRect            | 1280 (16) |           |           |           |           |           |           |           |  |
| SetWRefCon          | 44 (19)   |           |           |           |           |           |           |           |  |
| SetWTitle           | 67 (19)   | 143 (19)  | 916 (19)  | 1416 (37) | 2001 (37) |           |           |           |  |
| SFGetParms          | 869*(13)  | 349 (16)  | 427 (16)  | 2926*(16) |           |           |           |           |  |
| SFGetFile           | 450 (16)  |           |           |           |           |           |           |           |  |
| SFPPutFile          | 928 (17)  |           |           |           |           |           |           |           |  |
| SFPutParms          | 1347*(13) | 918 (17)  | 1399*(17) |           |           |           |           |           |  |
| SFReply             | 422 (16)  | 907 (17)  |           |           |           |           |           |           |  |
| SFTypeList          | 411 (16)  |           |           |           |           |           |           |           |  |
| shadow              | 1038 (26) |           |           |           |           |           |           |           |  |
| shadowed            | 398*(18)  | 439 (18)  | 483-(18)  | 484 (18)  | 496 (18)  |           |           |           |  |
| ShadowedFrame       | 2361*( 6) |           |           |           |           |           |           |           |  |
| shadowRgn           | 425*(18)  | 441-(18)  | 444-(18)  | 450 (18)  | 451 (18)  | 452 (18)  | 452 (18)  | 453 (18)  |  |
|                     | 458 (18)  | 459 (18)  |           |           |           |           |           |           |  |
| ShallowClone        | 82 (15)   | 64*(31)   | 157*(33)  | 190-(33)  | 191 (33)  | 216-(33)  | 228 (33)  |           |  |
| ShallowFree         | 73*(31)   | 238*(33)  | 255 (33)  |           |           |           |           |           |  |
| shouldOpenData      | 804*(17)  | 839-(17)  | 849 (17)  | 858 (17)  |           |           |           |           |  |
| shouldOpenRsrc      | 805*(17)  | 840-(17)  | 849 (17)  | 863 (17)  |           |           |           |           |  |
| Show                | 1657*(13) | 2000*(13) | 1648*(18) | 384 (19)  | 806 (19)  | 925*(19)  |           |           |  |
| ShowAWindow         | 1449*(17) | 1455 (17) |           |           |           |           |           |           |  |
| ShowControl         | 705 (21)  | 1324 (21) |           |           |           |           |           |           |  |
| ShowDocBeingPrinted | 452*(36)  | 1939 (37) | 1958 (37) | 1991*(37) |           |           |           |           |  |
| ShowError           | 1065*(13) | 206 (16)  | 396 (16)  | 1457 (16) | 1468 (16) | 2338 (16) | 2501 (16) | 2946*(16) |  |
| ShowReverted        | 1373*(13) | 1552*(13) | 310 (17)  | 338 (17)  | 1428*(17) | 1434 (17) | 1663*(18) | 1670 (18) |  |
|                     | 221*(38)  | 1452*(40) | 1456 (40) |           |           |           |           |           |  |
| ShowsOnScreen       | 434*(36)  | 1083 (37) | 2018*(37) | 2021-(37) | 2023-(37) |           |           |           |  |
| ShowWindow          | 757 (16)  | 930 (19)  |           |           |           |           |           |           |  |
| ShowWindows         | 1220*(13) | 2174 (16) | 2235 (16) | 1445*(17) |           |           |           |           |  |
| signature           | 2991*(13) | 2994*(13) | 253*(14)  | 260 (14)  | 1171*(14) | 1178 (14) |           |           |  |
| SignBit             | 111*( 6)  | 121 ( 6)  | 123 ( 6)  | 125 ( 6)  | 782*(26)  | 792 (26)  | 794 (26)  | 796 (26)  |  |
| SignedByte          | 1533 ( 6) | 1752 ( 6) | 1952 ( 6) | 179 (21)  | 322 (21)  | 710 (21)  | 137 (25)  | 258 (25)  |  |
|                     | 332 (25)  | 274 (26)  | 764 (26)  | 889 (26)  | 1688 (37) | 411 (40)  |           |           |  |
| SimpleStagger       | 2014*(13) | 156 (19)  | 941*(19)  |           |           |           |           |           |  |
| singleSelection     | 101*( 9)  | 183*( 9)  | 438*( 9)  | 531*( 9)  | 41*(10)   | 119 (10)  | 168 (10)  | 267-(10)  |  |
|                     | 2210*(10) | 2219 (10) | 2450*(10) | 2458 (10) |           |           |           |           |  |
| sipAlways           | 544*(13)  |           |           |           |           |           |           |           |  |
| sipAskUser          | 544*(13)  | 72 (17)   | 1155 (17) |           |           |           |           |           |  |
| SIPChoice           | 544*(13)  | 1136 (13) |           |           |           |           |           |           |  |
| sipNever            | 544*(13)  | 70 (17)   | 1140 (17) |           |           |           |           |           |  |
| SIZE                | 316 (27)  | 550 (28)  |           |           |           |           |           |           |  |
| Size                | 1517 (10) | 1617 (10) | 423 (12)  | 2189 (16) | 101 (18)  | 224 (27)  | 327 (27)  | 337 (27)  |  |
|                     | 341 (27)  | 44 (28)   | 46 (28)   | 54 (28)   | 225 (28)  | 226 (28)  | 393 (28)  | 597 (28)  |  |
|                     | 615 (28)  | 619 (28)  | 727 (28)  | 850 (28)  | 991 (28)  | 1059 (28) | 1061 (28) |           |  |
| size                | 79*( 9)   | 458 (10)  | 459 (10)  | 465 (10)  | 726 (10)  | 745 (10)  | 881 (10)  | 896 (10)  |  |
|                     | 896 (10)  | 978 (10)  | 993 (10)  | 993 (10)  | 1201 (10) | 1216 (10) | 1226 (10) | 1285 (10) |  |
|                     | 1300 (10) | 1310 (10) | 1360 (10) | 1362 (10) | 1391 (10) | 1393 (10) | 1543 (10) | 1551 (10) |  |
|                     | 1576-(10) | 1591-(10) | 1591 (10) | 1593-(10) | 1643 (10) | 1651 (10) | 1676-(10) | 1691-(10) |  |
|                     | 1691 (10) | 1693-(10) | 1812 (10) | 1813 (10) | 1819 (10) | 1894 (10) | 1920 (10) | 2086 (10) |  |
|                     | 2089 (10) | 2099 (10) | 2103 (10) | 2104 (10) | 2109 (10) | 2112 (10) | 2122 (10) | 2126 (10) |  |

|                       |           |           |           |           |           |           |           |           |
|-----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                       | 2127 (10) | 2698 (10) | 2699-(10) | 155*(20)  | 165-(20)  | 167 (20)  | 168 (20)  | 168 (20)  |
|                       | 170 (20)  | 172 (20)  | 172 (20)  | 56*(39)   | 64-(39)   | 66 (39)   | 75 (39)   |           |
| SizeControl           | 785 (21)  |           |           |           |           |           |           |           |
| SizeDeterminer        | 68 (5)    | 142 (5)   | 172 (5)   | 212 (5)   | 255 (5)   | 307 (5)   | 353 (5)   | 398 (5)   |
|                       | 447 (5)   | 514 (5)   | 255 (6)   | 781 (6)   | 862 (6)   | 990 (6)   | 1120 (6)  | 1391 (6)  |
|                       | 1614 (6)  | 1834 (6)  | 2025 (6)  | 2669 (6)  | 523*(13)  | 1466 (13) | 1480 (13) | 1770 (13) |
|                       | 2074 (13) | 2154 (13) | 2232 (13) | 2285 (13) | 180 (18)  | 225 (18)  | 225 (18)  | 14 (20)   |
|                       | 98 (21)   | 533 (21)  | 942 (21)  | 1206 (21) | 148 (38)  | 150 (38)  | 15 (40)   |           |
| sizeFillPages         | 528*(13)  | 650 (18)  |           |           |           |           |           |           |
| sizeFixed             | 619 (6)   | 619 (6)   | 2932 (6)  | 2932 (6)  | 219 (10)  | 2221 (10) | 2254 (10) | 530*(13)  |
| sizeLowSpaceReserve   | 635 (18)  | 286 (40)  | 374 (40)  |           |           |           |           |           |
| SIZEOF                | 393*(28)  | 481-(28)  | 497-(28)  | 497 (28)  |           |           |           |           |
|                       | 335 (6)   | 354 (6)   | 809 (6)   | 831 (6)   | 891 (6)   | 913 (6)   | 1018 (6)  | 1040 (6)  |
|                       | 1168 (6)  | 1190 (6)  | 1471 (6)  | 1489 (6)  | 1693 (6)  | 1711 (6)  | 1897 (6)  | 1915 (6)  |
|                       | 2129 (6)  | 2147 (6)  | 2721 (6)  | 2743 (6)  | 2964 (6)  | 2982 (6)  | 3431 (6)  | 3450 (6)  |
|                       | 246 (8)   | 229 (10)  | 249 (10)  | 2259 (10) | 2284 (10) | 12 (12)   | 742 (14)  | 764 (14)  |
|                       | 177 (15)  | 177 (15)  | 177 (15)  | 30 (16)   | 351 (16)  | 428 (16)  | 112 (18)  | 230 (18)  |
|                       | 245 (18)  | 160 (19)  | 183 (19)  | 71 (20)   | 89 (20)   | 151 (21)  | 175 (21)  | 981 (21)  |
|                       | 1000 (21) | 1322 (26) | 444 (28)  | 459 (28)  | 464 (28)  | 939 (28)  | 964 (28)  | 970 (28)  |
|                       | 325 (30)  | 86 (32)   | 112 (32)  | 109 (35)  | 151 (35)  | 185 (35)  | 1353 (37) | 1817 (37) |
|                       | 1947 (37) | 1950 (37) | 2110 (37) | 2113 (37) | 2114 (37) | 2167 (37) | 2179 (37) | 117 (40)  |
|                       | 137 (40)  | 1468 (40) | 742 (41)  | 749 (41)  | 751 (41)  | 756 (41)  | 757 (41)  |           |
| Sizeof                | 1571 (10) | 1585 (10) | 1671 (10) | 1685 (10) |           |           |           |           |
| sizeof                | 892 (10)  | 903 (10)  | 989 (10)  | 1000 (10) |           |           |           |           |
| sizePage              | 527*(13)  | 642 (18)  |           |           |           |           |           |           |
| sizePalette           | 2854*(13) | 832*(14)  | 852 (14)  | 859 (14)  | 864 (14)  |           |           |           |
| sizeRelSuperview      | 2461 (10) | 526*(13)  | 867 (14)  | 868 (14)  | 940 (14)  | 941 (14)  | 637 (18)  | 1720 (18) |
|                       | 374 (40)  |           |           |           |           |           |           |           |
| SizeResource          | 179 (28)  | 604 (28)  | 55 (32)   |           |           |           |           |           |
| sizeSuperview         | 525*(13)  | 639 (18)  | 1718 (18) | 374 (40)  | 376 (41)  |           |           |           |
| sizeTempReserve       | 393*(28)  | 480-(28)  | 487-(28)  | 487 (28)  | 496-(28)  | 496 (28)  |           |           |
| SizeVariable          | 41 (19)   | 41 (19)   | 229 (20)  | 230 (20)  | 241 (20)  | 242 (20)  |           |           |
| sizeVariable          | 102 (10)  | 102 (10)  | 529*(13)  | 633 (18)  | 26 (23)   | 26 (23)   | 376 (41)  |           |
| SizeWindow            | 325 (19)  | 617 (19)  | 824 (19)  |           |           |           |           |           |
| sizeX6                | 182*(30)  | 200 (30)  | 221 (30)  | 224 (30)  |           |           |           |           |
| sleep                 | 703*(13)  | 1119*(16) | 1142 (16) | 1152 (16) | 1160 (16) |           |           |           |
| slot                  | 947*(19)  | 979-(19)  | 981-(19)  | 983 (19)  | 988 (19)  | 989 (19)  |           |           |
| smHilite              | 497 (40)  |           |           |           |           |           |           |           |
| smKeyScript           | 754 (26)  | 495 (40)  |           |           |           |           |           |           |
| smScriptRight         | 495 (40)  |           |           |           |           |           |           |           |
| someMargins           | 1833*(37) | 1835-(37) | 1836 (37) |           |           |           |           |           |
| source                | 1073*(14) | 1079 (14) | 1080 (14) | 1082 (14) |           |           |           |           |
| SpaceForStyles        | 317*(38)  | 1128 (40) | 1464*(40) | 1472-(40) | 1477-(40) | 1605 (40) | 77 (41)   |           |
| SpaceIsLow            | 1066*(13) | 1782 (16) | 2955*(16) |           |           |           |           |           |
| spare                 | 27*(4)    |           |           |           |           |           |           |           |
| spoolFileName         | 300*(36)  | 588*(37)  | 591-(37)  | 1439*(37) | 1482 (37) | 1487 (37) | 1490 (37) |           |
| spoolMethod           | 1449*(37) | 1479-(37) | 1480 (37) |           |           |           |           |           |
| spoolVRefNum          | 301*(36)  | 588*(37)  | 592-(37)  | 1440*(37) | 1482 (37) | 1491 (37) |           |           |
| src1                  | 58*(42)   | 68*(42)   |           |           |           |           |           |           |
| src2                  | 58*(42)   | 68*(42)   |           |           |           |           |           |           |
| srcP                  | 552*(12)  | 559-(12)  | 560 (12)  | 564 (12)  | 566 (12)  | 572-(12)  | 572 (12)  |           |
| srcVPt                | 35*(42)   | 37*(42)   |           |           |           |           |           |           |
| StackTot              | 413*(28)  |           |           |           |           |           |           |           |
| stackTot              | 482-(28)  | 498-(28)  | 498 (28)  | 503 (28)  |           |           |           |           |
| stackVal              | 405*(28)  | 498 (28)  |           |           |           |           |           |           |
| stagger               | 1877*(13) | 155 (19)  | 197-(19)  |           |           |           |           |           |
| staggered             | 1890*(13) |           |           |           |           |           |           |           |
| start                 | 575*(9)   | 2557*(10) | 2563 (10) |           |           |           |           |           |
| startBreak            | 975*(37)  | 988-(37)  | 993-(37)  | 996 (37)  |           |           |           |           |
| startCell             | 252*(9)   | 294*(9)   | 311*(10)  | 315 (10)  | 316 (10)  | 320 (10)  | 693*(10)  | 709-(10)  |
|                       | 714 (10)  | 719 (10)  | 728 (10)  | 747 (10)  | 767*(10)  | 787 (10)  | 796 (10)  | 1081*(10) |
|                       | 1092 (10) | 1093 (10) |           |           |           |           |           |           |
| startChar             | 1544-(40) |           |           |           |           |           |           |           |
| StartEdit             | 616*(5)   | 522 (6)   | 3330*(6)  |           |           |           |           |           |
| startLoc              | 842*(37)  | 850-(37)  | 854 (37)  | 859 (37)  |           |           |           |           |
| StartMenuSetup        | 432*(30)  | 498 (30)  |           |           |           |           |           |           |
| startOfCurrentLine    | 401*(40)  | 470-(40)  | 499 (40)  | 500 (40)  | 501 (40)  | 503 (40)  | 504 (40)  | 508 (40)  |
|                       | 509 (40)  |           |           |           |           |           |           |           |
| startOfNextLine       | 402*(40)  | 472-(40)  | 474-(40)  | 500 (40)  | 504 (40)  | 511 (40)  |           |           |
| startOfSelection      | 399*(40)  | 422-(40)  | 429 (40)  | 435 (40)  | 436 (40)  | 461 (40)  | 466 (40)  | 494 (40)  |
|                       | 501 (40)  | 509 (40)  | 510 (40)  | 511 (40)  |           |           |           |           |
| StartSound            | 246 (8)   |           |           |           |           |           |           |           |
| state                 | 190*(5)   | 230*(5)   | 956*(6)   | 958 (6)   | 1085*(6)  | 1087 (6)  | 1653*(13) | 1657*(13) |
|                       | 2000*(13) | 2097*(13) | 2108*(13) | 2173*(13) | 2188*(13) | 1648*(18) | 1650 (18) | 1656 (18) |
|                       | 1782*(18) | 1784 (18) | 925*(19)  | 929 (19)  | 933 (19)  | 288*(21)  | 290 (21)  | 292 (21)  |
|                       | 364*(21)  | 366 (21)  | 368 (21)  | 659*(21)  | 663 (21)  | 664 (21)  | 668 (21)  | 760*(21)  |
|                       | 764 (21)  | 768 (21)  | 770 (21)  |           |           |           |           |           |
|                       | 258*(29)  | 577*(30)  | 581 (30)  |           |           |           |           |           |
| stateVariable         | 499 (5)   | 500*(5)   | 2721 (6)  | 2743 (6)  |           |           |           |           |
| StaticTextTemplate    | 499*(5)   | 2715 (6)  | 2735 (6)  | 2742 (6)  |           |           |           |           |
| StaticTextTemplatePtr | 2921*(13) | 223 (14)  | 406 (14)  | 1053 (14) | 1232*(14) | 752 (16)  | 2968 (16) | 742 (17)  |
| StdAlert              | 855 (40)  | 1098 (40) | 1471 (40) |           |           |           |           |           |
|                       | 396 (14)  | 350*(25)  | 859*(26)  |           |           |           |           |           |
| StdFieldToString      | 301*(19)  | 303 (19)  |           |           |           |           |           |           |
| stdHScreen            | 303*(19)  | 315 (19)  |           |           |           |           |           |           |
| stdScreen             | 880 (19)  |           |           |           |           |           |           |           |
| stdState              | 302*(19)  | 303 (19)  |           |           |           |           |           |           |
| stdVScreen            | 285 (38)  | 321 (38)  | 1037 (40) | 1485 (40) | 1488 (40) |           |           |           |
| STHandle              | 3271 (16) | 190 (39)  |           |           |           |           |           |           |
| StillDown             | 575*(9)   | 2557*(10) | 2563 (10) |           |           |           |           |           |
| stop                  | 252*(9)   | 294*(9)   | 312*(10)  | 317 (10)  | 318 (10)  | 320 (10)  | 694*(10)  | 710-(10)  |
| stopCell              | 715 (10)  | 719 (10)  | 728 (10)  | 747 (10)  | 767*(10)  | 787 (10)  | 796 (10)  | 1081*(10) |
|                       | 1094 (10) | 1095 (10) |           |           |           |           |           |           |
| StopEdit              | 618*(5)   | 516 (6)   | 3357*(6)  | 3363-(6)  | 3370-(6)  |           |           |           |
| storage               | 921*(13)  | 1210*(16) | 1276 (16) | 1278 (16) | 1291 (16) |           |           |           |
| stPtr                 | 2735*(6)  | 2742-(6)  | 2745 (6)  |           |           |           |           |           |
| str                   | 2911*(13) | 715*(14)  | 721*(14)  | 737-(14)  | 758 (14)  | 771 (14)  | 774 (14)  | 2647*(16) |
|                       | 2651-(16) | 2653-(16) | 2655-(16) | 2657-(16) | 2659-(16) | 2661-(16) | 2663-(16) | 2665-(16) |
|                       | 2669-(16) | 2672 (16) |           |           |           |           |           |           |



|                    |            |            |            |            |           |            |            |           |
|--------------------|------------|------------|------------|------------|-----------|------------|------------|-----------|
| Str255             | 33 (1)     | 38 (1)     | 67 (1)     | 67 (1)     | 69 (1)    | 69 (1)     | 73 (1)     | 75 (1)    |
|                    | 79 (1)     | 81 (1)     | 83 (1)     | 86 (1)     | 13 (2)    | 55 (2)     | 139 (2)    | 162 (2)   |
|                    | 190 (2)    | 236 (2)    | 236 (2)    | 267 (2)    | 295 (2)   | 318 (2)    | 318 (2)    | 355 (2)   |
|                    | 112 (5)    | 123 (5)    | 135 (5)    | 143 (5)    | 165 (5)   | 173 (5)    | 205 (5)    | 213 (5)   |
|                    | 244 (5)    | 275 (5)    | 281 (5)    | 284 (5)    | 330 (5)   | 376 (5)    | 421 (5)    | 480 (5)   |
|                    | 490 (5)    | 502 (5)    | 534 (5)    | 543 (5)    | 546 (5)   | 597 (5)    | 614 (5)    | 623 (5)   |
|                    | 665 (5)    | 688 (5)    | 633 (6)    | 754 (6)    | 782 (6)   | 822 (6)    | 863 (6)    | 904 (6)   |
|                    | 991 (6)    | 1031 (6)   | 1123 (6)   | 1181 (6)   | 1301 (6)  | 1348 (6)   | 1367 (6)   | 1590 (6)  |
|                    | 1810 (6)   | 2002 (6)   | 2342 (6)   | 2372 (6)   | 2428 (6)  | 2527 (6)   | 2561 (6)   | 2579 (6)  |
|                    | 2672 (6)   | 2734 (6)   | 2833 (6)   | 2877 (6)   | 2902 (6)  | 3134 (6)   | 3151 (6)   | 3281 (6)  |
|                    | 3284 (6)   | 3318 (6)   | 3360 (6)   | 3398 (6)   | 3418 (6)  | 3479 (6)   | 3496 (6)   | 3512 (6)  |
|                    | 3540 (6)   | 230 (7)    | 233 (7)    | 38 (8)     | 90 (8)    | 117 (8)    | 219 (8)    | 268 (8)   |
|                    | 114 (9)    | 398 (9)    | 469 (9)    | 495 (9)    | 552 (9)   | 588 (9)    | 623 (9)    | 660 (9)   |
|                    | 698 (9)    | 721 (9)    | 744 (9)    | 767 (9)    | 2138 (10) | 2274 (10)  | 2317 (10)  | 2389 (10) |
|                    | 2405 (10)  | 2611 (10)  | 2627 (10)  | 2761 (10)  | 2832 (10) | 3058 (10)  | 3093 (10)  | 3126 (10) |
|                    | 3161 (10)  | 240 (11)   | 246 (11)   | 247 (11)   | 586 (12)  | 620 (12)   | 625 (12)   | 651 (12)  |
|                    | 607 (13)   | 695 (13)   | 697 (13)   | 763 (13)   | 828 (13)  | 1157 (13)  | 1159 (13)  | 1192 (13) |
|                    | 1224 (13)  | 1231 (13)  | 1257 (13)  | 1286 (13)  | 1312 (13) | 1317 (13)  | 1333 (13)  | 1349 (13) |
|                    | 1398 (13)  | 1448 (13)  | 1731 (13)  | 1735 (13)  | 1855 (13) | 1883 (13)  | 2006 (13)  | 2031 (13) |
|                    | 2035 (13)  | 2057 (13)  | 2138 (13)  | 2155 (13)  | 2170 (13) | 2184 (13)  | 2196 (13)  | 2205 (13) |
|                    | 2270 (13)  | 2314 (13)  | 2344 (13)  | 2346 (13)  | 2466 (13) | 2578 (13)  | 2580 (13)  | 2651 (13) |
|                    | 2803 (13)  | 2813 (13)  | 2834 (13)  | 2837 (13)  | 2875 (13) | 2883 (13)  | 2883 (13)  | 2911 (13) |
|                    | 2988 (13)  | 2991 (13)  | 37 (14)    | 172 (14)   | 173 (14)  | 175 (14)   | 334 (14)   | 715 (14)  |
|                    | 721 (14)   | 1051 (14)  | 1063 (14)  | 1073 (14)  | 1073 (14) | 1155 (14)  | 1171 (14)  | 1242 (14) |
|                    | 1242 (14)  | 1266 (14)  | 39 (15)    | 177 (15)   | 270 (16)  | 906 (16)   | 1078 (16)  | 1955 (16) |
|                    | 2024 (16)  | 2031 (16)  | 2134 (16)  | 2536 (16)  | 2647 (16) | 2804 (16)  | 2805 (16)  | 136 (17)  |
|                    | 158 (17)   | 159 (17)   | 300 (17)   | 503 (17)   | 518 (17)  | 571 (17)   | 579 (17)   | 720 (17)  |
|                    | 736 (17)   | 758 (17)   | 759 (17)   | 903 (17)   | 909 (17)  | 1040 (17)  | 1219 (17)  | 1249 (17) |
|                    | 1289 (17)  | 1293 (17)  | 1380 (17)  | 1400 (17)  | 1463 (17) | 1466 (17)  | 1484 (17)  | 1103 (18) |
|                    | 1902 (18)  | 21 (19)    | 90 (19)    | 174 (19)   | 649 (19)  | 722 (19)   | 905 (19)   | 908 (19)  |
|                    | 1023 (19)  | 294 (20)   | 165 (21)   | 502 (21)   | 534 (21)  | 623 (21)   | 741 (21)   | 834 (21)  |
|                    | 922 (21)   | 1180 (21)  | 1430 (21)  | 119 (22)   | 134 (23)  | 282 (23)   | 38 (24)    | 120 (24)  |
|                    | 220 (25)   | 223 (25)   | 245 (25)   | 252 (25)   | 264 (25)  | 306 (25)   | 315 (25)   | 350 (25)  |
|                    | 355 (25)   | 356 (25)   | 390 (25)   | 406 (25)   | 413 (25)  | 430 (25)   | 442 (25)   | 445 (25)  |
|                    | 448 (25)   | 470 (25)   | 473 (25)   | 476 (25)   | 479 (25)  | 482 (25)   | 485 (25)   | 490 (25)  |
|                    | 494 (25)   | 498 (25)   | 50 (26)    | 172 (26)   | 186 (26)  | 233 (26)   | 253 (26)   | 331 (26)  |
|                    | 357 (26)   | 533 (26)   | 549 (26)   | 579 (26)   | 690 (26)  | 709 (26)   | 712 (26)   | 859 (26)  |
|                    | 877 (26)   | 903 (26)   | 910 (26)   | 923 (26)   | 1094 (26) | 1095 (26)  | 1182 (26)  | 1208 (26) |
|                    | 1232 (26)  | 1257 (26)  | 1280 (26)  | 1304 (26)  | 1330 (26) | 1353 (26)  | 1376 (26)  | 108 (28)  |
|                    | 156 (28)   | 303 (28)   | 331 (28)   | 759 (28)   | 802 (28)  | 207 (29)   | 248 (29)   | 148 (30)  |
|                    | 318 (30)   | 539 (30)   | 563 (30)   | 94 (31)    | 96 (31)   | 115 (31)   | 68 (33)    | 165 (33)  |
|                    | 264 (33)   | 152 (32)   | 155 (32)   | 255 (32)   | 289 (32)  | 261 (36)   | 300 (36)   | 466 (36)  |
|                    | 503 (36)   | 154 (37)   | 351 (37)   | 352 (37)   | 504 (37)  | 588 (37)   | 779 (37)   | 874 (37)  |
|                    | 1009 (37)  | 1372 (37)  | 1377 (37)  | 1439 (37)  | 1993 (37) | 2149 (37)  | 83 (38)    | 312 (38)  |
|                    | 356 (38)   | 448 (38)   | 532 (38)   | 586 (38)   | 129 (40)  | 1437 (40)  | 1633 (40)  | 161 (41)  |
|                    | 610 (41)   | 937 (41)   |            |            |           |            |            |           |
| Str2Dec            | 3524 (6)   |            |            |            |           |            |            |           |
| strID              | 733* (14)  | 744* (14)  | 754* (14)  | 758 (14)   |           |            |            |           |
| strIndex           | 253* (29)  | 561* (30)  | 565 (30)   |            |           |            |            |           |
| STRING             | 150 (8)    | 1103 (13)  | 2537 (16)  | 139 (25)   | 1318 (26) | 147 (36)   |            |           |
| String             | 199 (25)   |            |            |            |           |            |            |           |
| Strings8           | 147 (8)    | 148 (8)    | 61 (11)    | 590 (12)   | 602 (13)  | 2832 (13)  | 783 (14)   | 323 (15)  |
|                    | 123 (24)   | 139* (25)  | 140 (25)   | 292 (25)   | 369 (25)  | 434 (25)   | 334 (26)   | 335 (26)  |
|                    | 389 (26)   | 399 (26)   | 441 (26)   | 479 (26)   | 536 (26)  | 552 (26)   | 904 (26)   | 1127 (26) |
|                    | 1197 (26)  | 1343 (26)  | 1366 (26)  | 51 (31)    | 83 (31)   | 130 (31)   | 133 (31)   | 67 (33)   |
|                    | 164 (33)   | 166 (33)   | 167 (33)   | 105 (32)   | 191 (32)  | 241 (32)   | 243 (32)   | 274 (32)  |
|                    | 276 (32)   | 305 (32)   |            |            |           |            |            |           |
| StringHandle       | 30 (1)     | 31 (1)     | 251 (5)    | 1253 (6)   | 2820 (6)  | 2837 (6)   | 2882 (6)   | 47 (14)   |
|                    | 1161 (14)  | 897 (26)   | 156 (37)   |            |           |            |            |           |
| StringPtr          | 90 (18)    | 386 (25)   | 133 (26)   |            |           |            |            |           |
| StringToNum        | 3484 (6)   |            |            |            |           |            |            |           |
| StringWidth        | 1284 (6)   | 2382 (6)   | 371 (37)   | 814 (37)   | 818 (37)  |            |            |           |
| strip              | 1229* (37) | 1235 (37)  | 1236 (37)  | 1237 (37)  | 1239 (37) | 1848* (37) | 1852* (37) | 1855 (37) |
|                    | 1856 (37)  | 2034* (37) | 2037 (37)  | 2039 (37)  | 2040 (37) |            |            |           |
| StripAddress       | 34 (18)    |            |            |            |           |            |            |           |
| StripLong          | 31* (18)   | 34* (18)   | 68 (18)    | 68 (18)    | 108 (18)  | 109 (18)   |            |           |
| StripToPage        | 252* (36)  | 805 (37)   | 809 (37)   | 813 (37)   | 817 (37)  | 2031* (37) | 2039* (37) |           |
| StScrpHandle       | 40 (38)    | 385 (38)   | 391 (38)   | 520 (38)   | 494 (41)  | 551 (41)   |            |           |
| StuffHex           | 189 (37)   |            |            |            |           |            |            |           |
| StuffStyles        | 321* (38)  | 1485* (40) |            |            |           |            |            |           |
| StuffTRects        | 330* (38)  | 1273 (40)  | 1553* (40) |            |           |            |            |           |
| StuffText          | 325* (38)  | 1444 (40)  | 1518* (40) | 386 (41)   |           |            |            |           |
| Style              | 111 (9)    | 2054 (13)  | 346 (25)   | 842 (26)   | 893 (26)  | 265 (29)   | 596 (30)   | 80 (38)   |
| StyleItem          | 910 (26)   |            |            |            |           |            |            |           |
| styles             | 1520* (40) | 1543* (40) | 1544 (40)  |            |           |            |            |           |
| styleTab           | 1041 (40)  | 1499 (40)  | 1505* (40) |            |           |            |            |           |
| subbed             | 2805* (6)  | 2813* (6)  | 2822 (6)   |            |           |            |            |           |
| subJobFirstPage    | 325* (36)  | 1118* (37) | 1140 (37)  | 1144 (37)  | 1148 (37) | 1167 (37)  |            |           |
| subJobLastPage     | 1140 (37)  |            |            |            |           |            |            |           |
| subJobLastPage     | 325* (36)  | 1118* (37) | 1148 (37)  | 1167 (37)  |           |            |            |           |
| SubPt              | 881 (14)   | 958 (14)   | 485 (18)   | 314 (19)   | 315 (19)  | 321 (19)   | 358 (19)   | 959 (19)  |
|                    | 267 (21)   | 1056 (37)  | 1057 (37)  |            |           |            |            |           |
| SubstituteInTitle  | 2883* (13) | 1242* (14) | 1254* (14) | 1257* (14) | 2825 (16) | 1473 (17)  | 915 (19)   | 1403 (37) |
| SubViewChangedSize | 1570* (13) | 1817* (13) | 1601 (18)  | 1683* (18) | 651* (20) |            |            |           |
| SubViewMoved       | 1576* (13) | 1451 (18)  | 1691* (18) |            |           |            |            |           |
| subViewPt          | 1271* (18) | 1274* (18) | 1275 (18)  | 1276 (18)  | 1277 (18) | 1309* (18) | 1312* (18) | 1313 (18) |
|                    | 1314 (18)  | 1315 (18)  |            |            |           |            |            |           |
| SubVpt             | 3283 (16)  | 1701 (18)  | 1795 (18)  | 1822 (18)  | 366 (20)  | 663 (20)   | 37* (42)   |           |
| SUCC               | 2393 (6)   | 466 (40)   | 467 (40)   | 472 (40)   | 1544 (40) | 679 (41)   | 681 (41)   | 682 (41)  |
|                    | 721 (41)   | 761 (41)   | 795 (41)   |            |           |            |            |           |
| succeeded          | 703* (16)  |            |            |            |           |            |            |           |
| Success            | 37 (2)     | 222 (2)    | 288 (6)    | 333 (6)    | 1144 (6)  | 1420 (6)   | 1467 (6)   | 1643 (6)  |
|                    | 1689 (6)   | 1857 (6)   | 1893 (6)   | 2055 (6)   | 2111 (6)  | 2507 (6)   | 2694 (6)   | 225* (7)  |
|                    | 297* (8)   | 287 (12)   | 888 (14)   | 965 (14)   | 1005 (14) | 191 (15)   | 228 (16)   | 727 (16)  |
|                    | 1283 (16)  | 1420 (16)  | 1483 (16)  | 1513 (16)  | 2015 (16) | 2122 (16)  | 2176 (16)  | 2237 (16) |
|                    | 2293 (16)  | 2380 (16)  | 2475 (16)  | 2509 (16)  | 93 (17)   | 337 (17)   | 709 (17)   | 873 (17)  |
|                    | 1006 (17)  | 1178 (17)  | 1338 (17)  | 1360 (17)  | 1372 (17) | 211 (18)   | 454 (18)   | 73 (19)   |
|                    | 149 (19)   | 247 (20)   | 645 (21)   | 1234 (21)  | 1263 (21) | 219 (23)   | 308 (37)   | 643 (37)  |
|                    | 1173 (37)  | 1522 (37)  | 1550 (37)  | 1608 (37)  | 1734 (37) | 2082 (37)  | 2116 (37)  | 1169 (40) |

|                      |            |            |            |            |            |            |            |            |  |
|----------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| SuperToLocal         | 1238 (40)  | 84 (41)    | 385 (41)   | 507 (41)   | 657 (41)   | 853 (41)   |            |            |  |
| SuperViewChangedSize | 1506* (13) | 1801* (13) | 1275 (18)  | 1313 (18)  | 1699* (18) | 661* (20)  |            |            |  |
| SuperViewMoved       | 1568* (13) | 2573* (13) | 2767 (16)  | 364 (18)   | 1564 (18)  | 1708* (18) | 297* (23)  |            |  |
| Survey               | 1578* (13) | 1436 (18)  | 1738* (18) |            |            |            |            |            |  |
| SurveyEditText       | 721* (6)   | 741 (6)    |            |            |            |            |            |            |  |
| SwapClipViews        | 120* (5)   | 597 (6)    | 713* (6)   |            |            |            |            |            |  |
| switchingIn          | 1055* (13) | 801 (16)   | 2978* (16) |            |            |            |            |            |  |
| swMode               | 1660* (16) | 1672- (16) | 1675 (16)  | 1683 (16)  | 1690 (16)  | 1692 (16)  |            |            |  |
| sym                  | 235 (8)    |            |            |            |            |            |            |            |  |
| SynchView            | 602* (13)  | 2832* (13) | 783* (14)  | 786 (14)   | 323* (15)  | 326 (15)   |            |            |  |
|                      | 246* (38)  | 358 (40)   | 1277 (40)  | 1426 (40)  | 1574* (40) | 255 (41)   | 287 (41)   | 558 (41)   |  |
|                      | 860 (41)   |            |            |            |            |            |            |            |  |
| synthRec             | 225* (8)   | 233 (8)    | 246 (8)    | 246 (8)    |            |            |            |            |  |
| SysBeep              | 411 (6)    | 248 (8)    | 1578 (16)  | 732 (40)   |            |            |            |            |  |
| SysEnvirons          | 109 (26)   |            |            |            |            |            |            |            |  |
| SysEnvRec            | 94 (26)    |            |            |            |            |            |            |            |  |
| SysFontFam           | 360* (26)  | 366 (26)   |            |            |            |            |            |            |  |
| SysFontSize          | 470* (14)  | 686 (14)   | 61* (26)   | 65 (26)    |            |            |            |            |  |
| SystemClick          | 1605 (16)  |            |            |            |            |            |            |            |  |
| SystemEdit           | 1994 (16)  |            |            |            |            |            |            |            |  |
| systemFont           | 687 (14)   | 261 (26)   | 363 (26)   |            |            |            |            |            |  |
| SystemTask           | 1185 (16)  |            |            |            |            |            |            |            |  |
| systemVersion        | 174* (25)  | 122 (26)   |            |            |            |            |            |            |  |
| sysVRefNum           | 180* (25)  |            |            |            |            |            |            |            |  |
| sysWindowAct         | 1554* (16) | 2393* (16) | 2409- (16) | 2411 (16)  | 2413 (16)  |            |            |            |  |
| sz                   | 301* (28)  | 305- (28)  | 306 (28)   | 308 (28)   |            |            |            |            |  |
| szCodeReserve        | 316* (27)  | 550* (28)  | 553- (28)  |            |            |            |            |            |  |
| szMemReserve         | 316* (27)  | 550* (28)  | 554- (28)  |            |            |            |            |            |  |
| <b>-T-</b>           |            |            |            |            |            |            |            |            |  |
| TlPBlock             | 26 (35)    | 27* (35)   | 185 (35)   |            |            |            |            |            |  |
| TlPBlockPtr          | 26* (35)   | 181 (35)   | 185 (35)   |            |            |            |            |            |  |
| table                | 729* (14)  | 739- (14)  | 740 (14)   | 742 (14)   | 749 (14)   |            |            |            |  |
| tableOffset          | 731* (14)  | 745- (14)  | 749 (14)   | 764- (14)  | 764 (14)   |            |            |            |  |
| TailIPatch           | 143 (4)    | 146 (4)    | 95* (34)   | 177* (35)  | 202- (35)  |            |            |            |  |
| TailPatch            | 144 (4)    | 154 (4)    | 155 (4)    | 456 (14)   | 85* (34)   | 143* (35)  | 167- (35)  |            |  |
| TakeEvent            | 1337* (16) | 1339- (16) | 1344 (16)  |            |            |            |            |            |  |
| TAlias               | 865* (26)  | 902 (26)   |            |            |            |            |            |            |  |
| TApplication         | 682* (13)  | 687* (13)  | 695* (13)  | 697* (13)  | 703* (13)  | 708* (13)  | 712* (13)  | 714* (13)  |  |
|                      | 716* (13)  | 721* (13)  | 723* (13)  | 728* (13)  | 732* (13)  | 737* (13)  | 741* (13)  | 744* (13)  |  |
|                      | 748* (13)  | 754* (13)  | 758* (13)  | 763* (13)  | 768* (13)  | 770* (13)  | 774* (13)  | 777* (13)  |  |
|                      | 780* (13)  | 784* (13)  | 787* (13)  | 794* (13)  | 799* (13)  | 804* (13)  | 812* (13)  | 816* (13)  |  |
|                      | 822* (13)  | 828* (13)  | 833* (13)  | 838* (13)  | 842* (13)  | 852* (13)  | 859* (13)  | 865* (13)  |  |
|                      | 869* (13)  | 887* (13)  | 894* (13)  | 897* (13)  | 899* (13)  | 902* (13)  | 904* (13)  | 907* (13)  |  |
|                      | 910* (13)  | 916* (13)  | 921* (13)  | 927* (13)  | 933* (13)  | 937* (13)  | 943* (13)  | 946* (13)  |  |
|                      | 951* (13)  | 957* (13)  | 962* (13)  | 969* (13)  | 971* (13)  | 976* (13)  | 994* (13)  | 997* (13)  |  |
|                      | 1002* (13) | 1006* (13) | 1012* (13) | 1016* (13) | 1023* (13) | 1027* (13) | 1032* (13) | 1035* (13) |  |
|                      | 1040* (13) | 1044* (13) | 1049* (13) | 1052* (13) | 1055* (13) | 1061* (13) | 1065* (13) | 1066* (13) |  |
|                      | 2594 (13)  | 58* (16)   | 173* (16)  | 190* (16)  | 239* (16)  | 252* (16)  | 261* (16)  | 270* (16)  |  |
|                      | 329* (16)  | 375* (16)  | 408* (16)  | 468* (16)  | 499* (16)  | 540* (16)  | 560* (16)  | 577* (16)  |  |
|                      | 607* (16)  | 618* (16)  | 627* (16)  | 670* (16)  | 684* (16)  | 701* (16)  | 839* (16)  | 872* (16)  |  |
|                      | 881* (16)  | 906* (16)  | 1021* (16) | 1051* (16) | 1060* (16) | 1078* (16) | 1090* (16) | 1119* (16) |  |
|                      | 1210* (16) | 1303* (16) | 1328* (16) | 1352* (16) | 1379* (16) | 1430* (16) | 1524* (16) | 1547* (16) |  |
|                      | 1644* (16) | 1656* (16) | 1703* (16) | 1731* (16) | 1745* (16) | 1813* (16) | 1823* (16) | 1834* (16) |  |
|                      | 1871* (16) | 1883* (16) | 1892* (16) | 1902* (16) | 1921* (16) | 1935* (16) | 1948* (16) | 2024* (16) |  |
|                      | 2130* (16) | 2183* (16) | 2244* (16) | 2318* (16) | 2390* (16) | 2430* (16) | 2486* (16) | 2518* (16) |  |
|                      | 2532* (16) | 2645* (16) | 2681* (16) | 2734* (16) | 2746* (16) | 2787* (16) | 2801* (16) | 2839* (16) |  |
|                      | 2926* (16) | 2946* (16) | 2955* (16) | 2978* (16) | 2998* (16) | 3098* (16) | 3361* (16) | 3375* (16) |  |
| target               | 311* (30)  | 345- (30)  |            |            |            |            |            |            |  |
| targetID             | 1882* (13) | 131 (19)   | 201- (19)  |            |            |            |            |            |  |
| TAssociation         | 59* (1)    | 63* (1)    | 65* (1)    | 67* (1)    | 69* (1)    | 71* (1)    | 73* (1)    | 75* (1)    |  |
|                      | 77* (1)    | 79* (1)    | 81* (1)    | 83* (1)    | 86* (1)    | 91* (2)    | 109* (2)   | 129* (2)   |  |
|                      | 139* (2)   | 162* (2)   | 181* (2)   | 190* (2)   | 236* (2)   | 267* (2)   | 295* (2)   | 318* (2)   |  |
|                      | 355* (2)   | 59 (5)     | 258 (6)    | 301 (6)    | 2751 (13)  |            |            |            |  |
| TButton              | 138* (5)   | 141* (5)   | 147* (5)   | 151* (5)   | 153* (5)   | 17 (6)     | 780* (6)   | 796* (6)   |  |
|                      | 819* (6)   | 844* (6)   |            |            |            |            |            |            |  |
| TCellSelectCommand   | 672* (9)   | 677* (9)   | 681* (9)   | 686* (9)   | 690* (9)   | 696* (9)   | 625 (10)   | 2853* (10) |  |
|                      | 2869* (10) | 2930* (10) | 3028* (10) | 3058* (10) |            |            |            |            |  |
| TCheckBox            | 168* (5)   | 171* (5)   | 177* (5)   | 181* (5)   | 183* (5)   | 186* (5)   | 188* (5)   | 190* (5)   |  |
|                      | 192* (5)   | 194* (5)   | 18 (6)     | 861* (6)   | 878* (6)   | 901* (6)   | 924* (6)   | 936* (6)   |  |
|                      | 947* (6)   | 956* (6)   | 965* (6)   | 974* (6)   |            |            |            |            |  |
| TCluster             | 247* (5)   | 254* (5)   | 260* (5)   | 264* (5)   | 266* (5)   | 269* (5)   | 271* (5)   | 273* (5)   |  |
|                      | 275* (5)   | 277* (5)   | 279* (5)   | 281* (5)   | 284* (5)   | 20 (6)     | 1119* (6)  | 1158* (6)  |  |
|                      | 1178* (6)  | 1200* (6)  | 1212* (6)  | 1222* (6)  | 1245* (6)  | 1301* (6)  | 1313* (6)  | 1324* (6)  |  |
|                      | 1348* (6)  | 1367* (6)  |            |            |            |            |            |            |  |
| TColSelectCommand    | 734* (9)   | 737* (9)   | 742* (9)   | 627 (10)   | 3112* (10) | 3126* (10) |            |            |  |
| TCommand             | 92 (5)     | 96 (5)     | 104 (5)    | 106 (5)    | 470 (5)    | 584 (5)    | 586 (5)    | 589 (5)    |  |
|                      | 474 (6)    | 494 (6)    | 570 (6)    | 588 (6)    | 2240 (6)   | 3018 (6)   | 3021 (6)   | 3039 (6)   |  |
|                      | 3042 (6)   | 3062 (6)   | 3065 (6)   | 224 (9)    | 639* (9)   | 692 (9)    | 620 (10)   | 3030 (10)  |  |
|                      | 616 (13)   | 634 (13)   | 637 (13)   | 642 (13)   | 708 (13)   | 714 (13)   | 723 (13)   | 728 (13)   |  |
|                      | 732 (13)   | 737 (13)   | 742 (13)   | 742 (13)   | 744 (13)   | 748 (13)   | 754 (13)   | 943 (13)   |  |
|                      | 946 (13)   | 971 (13)   | 994 (13)   | 1404 (13)  | 1623 (13)  | 1626 (13)  | 1702 (13)  | 1990 (13)  |  |
|                      | 2100 (13)  | 2176 (13)  | 2249 (13)  | 2305 (13)  | 2393 (13)  | 2430* (13) | 2461* (13) | 2466* (13) |  |
|                      | 2473* (13) | 2474* (13) | 2475* (13) | 2476* (13) | 2480* (13) | 2486* (13) | 2492* (13) | 2494 (13)  |  |
|                      | 2519* (13) | 2528* (13) | 2536 (13)  | 2692 (13)  | 2724 (13)  | 201 (15)   | 214 (15)   | 233 (15)   |  |
|                      | 245 (15)   | 627 (16)   | 670 (16)   | 701 (16)   | 1303 (16)  | 1328 (16)  | 1331 (16)  | 1352 (16)  |  |
|                      | 1383 (16)  | 1524 (16)  | 1547 (16)  | 1555 (16)  | 1644 (16)  | 1656 (16)  | 1703 (16)  | 1948 (16)  |  |
|                      | 1953 (16)  | 2244 (16)  | 3098 (16)  | 3098 (16)  | 3101 (16)  | 3173 (16)  | 3218 (16)  | 299 (17)   |  |
|                      | 788 (17)   | 784 (18)   | 803 (18)   | 1300 (18)  | 687 (19)   | 77 (21)    | 303 (21)   | 677 (21)   |  |
|                      | 1049 (21)  | 1342 (21)  | 12* (22)   | 34* (22)   | 44* (22)   | 53* (22)   | 62* (22)   | 71* (22)   |  |
|                      | 81* (22)   | 97* (22)   | 99 (22)    | 109* (22)  | 119* (22)  | 160 (24)   | 328 (36)   | 339 (36)   |  |
|                      | 401 (36)   | 445 (36)   | 476* (36)  | 654 (37)   | 1120 (37)  | 1316 (37)  | 1427 (37)  | 179 (38)   |  |
|                      | 192 (38)   | 196 (38)   | 368* (38)  | 695 (40)   | 828 (40)   | 830 (40)   | 884 (40)   |            |  |
| TControl             | 247* (5)   | 299* (5)   | 345* (5)   | 391* (5)   | 437* (5)   | 505* (5)   | 15 (6)     | 53 (6)     |  |
|                      | 460 (6)    | 460 (6)    | 480 (6)    | 572 (6)    | 578 (6)    | 2060* (13) | 2073* (13) | 2078* (13) |  |
|                      | 2082* (13) | 2084* (13) | 2087* (13) | 2089* (13) | 2091* (13) | 2093* (13) | 2095* (13) | 2097* (13) |  |
|                      | 2099* (13) | 2102* (13) | 2104* (13) | 2106* (13) | 2108* (13) | 2110* (13) | 2112* (13) | 2114* (13) |  |
|                      | 2116* (13) | 2118* (13) | 2120* (13) | 2122* (13) | 2128* (13) | 2131* (13) | 2135* (13) | 2138* (13) |  |
|                      | 2148* (13) | 2530 (13)  | 2532 (13)  | 36 (21)    | 97* (21)   | 127* (21)  | 162* (21)  | 204* (21)  |  |

|                   |           |           |           |           |           |           |           |           |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                   | 216*(21)  | 226*(21)  | 248*(21)  | 260*(21)  | 274*(21)  | 288*(21)  | 302*(21)  | 317*(21)  |
|                   | 337*(21)  | 352*(21)  | 364*(21)  | 378*(21)  | 393*(21)  | 404*(21)  | 415*(21)  | 424*(21)  |
|                   | 438*(21)  | 451*(21)  | 460*(21)  | 469*(21)  | 492*(21)  | 502*(21)  |           |           |
| TControlTracker   | 2528*(13) | 2532*(13) | 2534*(13) | 2538*(13) | 2541*(13) | 36*(21)   | 54*(21)   | 65*(21)   |
|                   | 75*(21)   | 305(21)   |           |           |           |           |           |           |
| TctlMgr           | 138*(5)   | 168*(5)   | 208*(5)   | 16(6)     | 2148*(13) | 2153*(13) | 2159*(13) | 2161*(13) |
|                   | 2163*(13) | 2166*(13) | 2168*(13) | 2170*(13) | 2173*(13) | 2175*(13) | 2178*(13) | 2180*(13) |
|                   | 2182*(13) | 2184*(13) | 2186*(13) | 2188*(13) | 2190*(13) | 2192*(13) | 2194*(13) | 2196*(13) |
|                   | 2198*(13) | 2200*(13) | 2202*(13) | 2205*(13) | 2221*(13) | 532*(21)  | 552*(21)  | 567*(21)  |
|                   | 579*(21)  | 590*(21)  | 602*(21)  | 623*(21)  | 659*(21)  | 676*(21)  | 690*(21)  | 721*(21)  |
|                   | 731*(21)  | 741*(21)  | 750*(21)  | 760*(21)  | 777*(21)  | 798*(21)  | 816*(21)  | 834*(21)  |
|                   | 852*(21)  | 870*(21)  | 890*(21)  | 922*(21)  |           |           |           |           |
| TDeskScrapView    | 2550*(13) | 2561*(13) | 2565*(13) | 2568*(13) | 2570*(13) | 2571*(13) | 2572*(13) | 2573*(13) |
|                   | 2574*(13) | 2578*(13) | 2580*(13) | 2622(13)  | 477(14)   | 15*(23)   | 42*(23)   | 63*(23)   |
|                   | 74*(23)   | 134*(23)  | 153*(23)  | 229*(23)  | 282*(23)  | 297*(23)  | 309*(23)  |           |
| TDialogTEView     | 61(5)     | 110(5)    | 616(5)    | 673*(5)   | 677*(5)   | 679*(5)   | 681*(5)   | 683*(5)   |
|                   | 685*(5)   | 688*(5)   | 259(6)    | 302(6)    | 611(6)    | 614(6)    | 2550*(6)  | 2561*(6)  |
|                   | 2576*(6)  | 2616*(6)  | 2636*(6)  | 2648*(6)  | 3330(6)   |           |           |           |
| TDialogView       | 56*(5)    | 66*(5)    | 73*(5)    | 77*(5)    | 79*(5)    | 82*(5)    | 84*(5)    | 86*(5)    |
|                   | 88*(5)    | 90*(5)    | 92*(5)    | 95*(5)    | 98*(5)    | 100*(5)   | 102*(5)   | 104*(5)   |
|                   | 106*(5)   | 108*(5)   | 110*(5)   | 112*(5)   | 114*(5)   | 116*(5)   | 118*(5)   | 120*(5)   |
|                   | 123*(5)   | 14(6)     | 253*(6)   | 298*(6)   | 345*(6)   | 367*(6)   | 379*(6)   | 392*(6)   |
|                   | 408*(6)   | 420*(6)   | 435*(6)   | 447*(6)   | 474*(6)   | 493*(6)   | 509*(6)   | 538*(6)   |
|                   | 561*(6)   | 570*(6)   | 588*(6)   | 611*(6)   | 633*(6)   | 643*(6)   | 667*(6)   | 698*(6)   |
|                   | 713*(6)   | 754*(6)   | 1254(6)   | 1280(6)   | 2783(6)   | 2792(6)   | 3315(6)   |           |
| TDocument         | 66(5)     | 73(5)     | 147(5)    | 177(5)    | 217(5)    | 260(5)    | 312(5)    | 358(5)    |
|                   | 403(5)    | 452(5)    | 519(5)    | 573(5)    | 649(5)    | 253(6)    | 298(6)    | 796(6)    |
|                   | 878(6)    | 1006(6)   | 1158(6)   | 1433(6)   | 1656(6)   | 1870(6)   | 2080(6)   | 2708(6)   |
|                   | 2951(6)   | 3415(6)   | 172(9)    | 193(9)    | 428(9)    | 451(9)    | 523(9)    | 30(10)    |
|                   | 140(10)   | 2200(10)  | 2234(10)  | 2442(10)  | 650(13)   | 652(13)   | 829(13)   | 838(13)   |
|                   | 894(13)   | 899(13)   | 907(13)   | 1077*(13) | 1140*(13) | 1143*(13) | 1148*(13) | 1153*(13) |
|                   | 1157*(13) | 1159*(13) | 1165*(13) | 1170*(13) | 1174*(13) | 1179*(13) | 1181*(13) | 1190*(13) |
|                   | 1192*(13) | 1198*(13) | 1199(13)  | 1213*(13) | 1220*(13) | 1224*(13) | 1231*(13) | 1238*(13) |
|                   | 1243*(13) | 1248*(13) | 1257*(13) | 1265*(13) | 1277*(13) | 1284*(13) | 1292*(13) | 1301*(13) |
|                   | 1302(13)  | 1312*(13) | 1316*(13) | 1332*(13) | 1347*(13) | 1363*(13) | 1372*(13) | 1373*(13) |
|                   | 1379*(13) | 1391*(13) | 1398*(13) | 1404*(13) | 1405*(13) | 1409*(13) | 1412*(13) | 1415*(13) |
|                   | 1418*(13) | 1421*(13) | 1426*(13) | 1463(13)  | 1475(13)  | 1482(13)  | 1776(13)  | 1931(13)  |
|                   | 1936(13)  | 2026(13)  | 2078(13)  | 2159(13)  | 2237(13)  | 2291(13)  | 2434(13)  | 2461(13)  |
|                   | 2565(13)  | 2643(13)  | 2852(13)  | 2861(13)  | 2865(13)  | 2871(13)  | 831(14)   | 899(14)   |
|                   | 975(14)   | 1020(14)  | 57(15)    | 105(15)   | 252(16)   | 270(16)   | 278(16)   | 284(16)   |
|                   | 514(16)   | 607(16)   | 684(16)   | 704(16)   | 1051(16)  | 1064(16)  | 1436(16)  | 2132(16)  |
|                   | 2186(16)  | 2187(16)  | 2432(16)  | 10*(17)   | 101*(17)  | 125*(17)  | 135*(17)  | 146*(17)  |
|                   | 156*(17)  | 181*(17)  | 222*(17)  | 232*(17)  | 242*(17)  | 265*(17)  | 275*(17)  | 290*(17)  |
|                   | 299*(17)  | 352*(17)  | 365*(17)  | 388*(17)  | 410*(17)  | 436*(17)  | 454*(17)  | 464*(17)  |
|                   | 473*(17)  | 491*(17)  | 503*(17)  | 515*(17)  | 571*(17)  | 605*(17)  | 615*(17)  | 720*(17)  |
|                   | 734*(17)  | 734(17)   | 754*(17)  | 784*(17)  | 799*(17)  | 901*(17)  | 913(17)   | 957*(17)  |
|                   | 1036*(17) | 1058(17)  | 1206*(17) | 1207(17)  | 1219*(17) | 1247*(17) | 1287*(17) | 1380*(17) |
|                   | 1399*(17) | 1428*(17) | 1445*(17) | 1463*(17) | 1484*(17) | 179(18)   | 219(18)   | 15(19)    |
|                   | 83(19)    | 719(19)   | 46(20)    | 127(21)   | 552(21)   | 962(21)   | 1244(21)  | 12(22)    |
|                   | 42(23)    | 95(36)    | 192(36)   | 245(37)   | 319(37)   | 142(38)   | 160(38)   | 13(40)    |
|                   | 70(40)    |           |           |           |           |           |           |           |
| TEActivate        | 1018(40)  | 1187(40)  |           |           |           |           |           |           |
| TEAutoView        | 207(40)   |           |           |           |           |           |           |           |
| TECalText         | 99(23)    | 1250(40)  |           |           |           |           |           |           |
| TEClick           | 891(40)   |           |           |           |           |           |           |           |
| TEContinuousStyle | 36*(38)   | 555(40)   |           |           |           |           |           |           |
| TECustomHook      | 44*(38)   |           |           |           |           |           |           |           |
| TEDeactivate      | 1002(40)  | 1192(40)  |           |           |           |           |           |           |
| TEDelete          | 112(41)   | 193(41)   |           |           |           |           |           |           |
| TEDispose         | 3210(6)   | 107(23)   | 191(40)   |           |           |           |           |           |
| TEEditText        | 60(5)     | 86(5)     | 98(5)     | 100(5)    | 120(5)    | 561*(5)   | 568*(5)   | 573*(5)   |
|                   | 577*(5)   | 579*(5)   | 582*(5)   | 584*(5)   | 586*(5)   | 588*(5)   | 591*(5)   | 593*(5)   |
|                   | 595*(5)   | 597*(5)   | 599*(5)   | 602*(5)   | 604*(5)   | 606*(5)   | 608*(5)   | 610*(5)   |
|                   | 612*(5)   | 614*(5)   | 616*(5)   | 618*(5)   | 620*(5)   | 623*(5)   | 638*(5)   | 675(5)    |
|                   | 679(5)    | 25(6)     | 408(6)    | 453(6)    | 457(6)    | 509(6)    | 538(6)    | 546(6)    |
|                   | 547(6)    | 591(6)    | 592(6)    | 593(6)    | 594(6)    | 705(6)    | 706(6)    | 713(6)    |
|                   | 721(6)    | 2576(6)   | 2925*(6)  | 2951*(6)  | 2974*(6)  | 2995*(6)  | 3007*(6)  | 3018*(6)  |
|                   | 3039*(6)  | 3061*(6)  | 3088*(6)  | 3102*(6)  | 3113*(6)  | 3134*(6)  | 3151*(6)  | 3175*(6)  |
|                   | 3218*(6)  | 3231*(6)  | 3244*(6)  | 3256*(6)  | 3267*(6)  | 3281*(6)  | 3312*(6)  | 3330*(6)  |
|                   | 3357*(6)  | 3379*(6)  |           |           |           |           |           |           |
| teForceLeft       | 3196(6)   |           |           |           |           |           |           |           |
| TEGetHeight       | 314(40)   |           |           |           |           |           |           |           |
| TEGetPoint        | 435(40)   | 438(40)   |           |           |           |           |           |           |
| TEGetStyle        | 436(40)   | 737(41)   |           |           |           |           |           |           |
| TEHandle          | 3181(6)   | 79(23)    | 37(38)    | 41(38)    | 44(38)    | 47(38)    | 96(38)    | 374(38)   |
|                   | 605(38)   | 613(38)   | 52(39)    | 163(39)   | 1204(40)  |           |           |           |
| TEIdle            | 682(40)   |           |           |           |           |           |           |           |
| TEInit            | 389(14)   |           |           |           |           |           |           |           |
| TEInsert          | 144(41)   | 233(41)   |           |           |           |           |           |           |
| TEIntHook         | 34*(38)   | 44(38)    |           |           |           |           |           |           |
| TEJustCenter      | 1288(6)   | 520(40)   |           |           |           |           |           |           |
| teJustLeft        | 620(6)    | 2639(6)   | 2688(6)   | 3186(6)   | 3197(6)   | 3202(6)   | 245(23)   | 466(40)   |
|                   | 515(40)   | 378(41)   |           |           |           |           |           |           |
| teJustRight       | 2350(6)   | 517(40)   |           |           |           |           |           |           |
| TEKey             | 710(40)   | 858(41)   |           |           |           |           |           |           |
| teLength          | 64(39)    | 185-(40)  | 306(40)   | 307(40)   | 421(40)   | 865(40)   | 966(40)   | 1537-(40) |
|                   | 1544(40)  |           |           |           |           |           |           |           |
| temp              | 1281*(37) | 1301-(37) | 1303(37)  |           |           |           |           |           |
| tempClipView      | 2980*(16) | 2982-(16) | 2985(16)  | 2986(16)  |           |           |           |           |
| tempCPort         | 1837*(18) | 1843(18)  | 1892(18)  |           |           |           |           |           |
| template          | 2875*(13) | 1063*(14) | 1087(14)  | 1094(14)  | 1097(14)  | 1099(14)  | 1104(14)  | 1105(14)  |
| templateHandle    | 1228*(16) | 1254-(16) | 1255(16)  | 1256(16)  | 1258(16)  | 1280(16)  |           |           |
| tempPort          | 1836*(18) | 1845(18)  | 1894(18)  |           |           |           |           |           |
| tempRect          | 554*(19)  |           |           |           |           |           |           |           |
| tempString        | 536*(26)  | 539(26)   | 540(26)   | 552*(26)  | 559(26)   | 560(26)   | 1343*(26) | 1345(26)  |
|                   | 1346(26)  | 1366*(26) | 1368(26)  | 1369(26)  |           |           |           |           |
| TENew             | 3200(6)   | 91(23)    | 1227(40)  |           |           |           |           |           |
| TEntriesList      | 49*(1)    | 51*(1)    | 61(1)     | 73*(2)    | 94(2)     |           |           |           |
| TEntry            | 28*(1)    | 33*(1)    | 35*(1)    | 38*(1)    | 71(1)     | 73(1)     | 75(1)     | 77(1)     |

|                         |            |            |            |            |            |            |            |            |
|-------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                         | 77 ( 1)    | 13* ( 2)   | 43* ( 2)   | 55* ( 2)   | 76 ( 2)    | 76 ( 2)    | 78 ( 2)    | 78 ( 2)    |
|                         | 114 ( 2)   | 129 ( 2)   | 139 ( 2)   | 147 ( 2)   | 149 ( 2)   | 156 ( 2)   | 162 ( 2)   | 167 ( 2)   |
|                         | 181 ( 2)   | 181 ( 2)   | 184 ( 2)   | 193 ( 2)   | 209 ( 2)   | 239 ( 2)   | 244 ( 2)   | 270 ( 2)   |
|                         | 278 ( 2)   | 280 ( 2)   | 287 ( 2)   | 298 ( 2)   | 303 ( 2)   | 321 ( 2)   | 329 ( 2)   | 331 ( 2)   |
|                         | 338 ( 2)   | 680 ( 6)   | 1157 (14)  | 59 (15)    |            |            |            |            |
| TENumStyles             | 47* (38)   | 1468 (40)  |            |            |            |            |            |            |
| Terminate               | 597* (13)  | 96 (14)    | 337* (15)  | 531 (16)   | 204* (36)  | 2048* (37) |            |            |
| TESelView               | 2655 ( 6)  | 1340 (40)  |            |            |            |            |            |            |
| TESetJust               | 1350 (40)  |            |            |            |            |            |            |            |
| TESetSelect             | 3248 ( 6)  | 3272 ( 6)  | 865 (40)   | 192 (41)   | 205 (41)   | 219 (41)   |            |            |
| TESetStyle              | 1379 (40)  |            |            |            |            |            |            |            |
| TESetText               | 3208 ( 6)  |            |            |            |            |            |            |            |
| test                    | 106* (30)  | 122* (30)  | 123 (30)   | 131 (30)   | 133 (30)   |            |            |            |
| TestControl             | 680 (21)   | 1055 (21)  | 1357 (21)  |            |            |            |            |            |
| TestDoc                 | 284* (16)  | 311 (16)   |            |            |            |            |            |            |
| TestEntry               | 77* ( 1)   | 181* ( 2)  | 184 ( 2)   |            |            |            |            |            |
| TestEvtHandler          | 805* (13)  | 1022* (16) | 1037 (16)  |            |            |            |            |            |
| TestForError            | 590* (26)  | 610 (26)   | 621 (26)   | 628 (26)   | 637 (26)   | 638 (26)   | 642 (26)   |            |
| TestItem                | 132* (11)  | 154* (11)  | 283* (11)  | 313* (12)  | 327 (12)   | 510* (12)  | 524 (12)   | 726* (12)  |
|                         | 738 (12)   | 756 (12)   |            |            |            |            |            |            |
| TestKey                 | 144* ( 2)  | 148* ( 2)  | 150* ( 2)  | 152* ( 2)  | 156 ( 2)   | 209* ( 2)  | 212* ( 2)  | 275* ( 2)  |
|                         | 279* ( 2)  | 283* ( 2)  | 287 ( 2)   |            |            |            |            |            |
| Testkey                 | 281* ( 2)  |            |            |            |            |            |            |            |
| TestMouse               | 1269* (18) | 1278* (18) | 1284 (18)  | 1307* (18) | 1315* (18) | 1318* (18) | 1327 (18)  |            |
| TestSubID               | 955* (18)  | 961 (18)   | 962* (18)  | 971 (18)   |            |            |            |            |
| TestSubview             | 1526* (13) | 988* (18)  | 991 (18)   | 1390* (18) | 1393 (18)  |            |            |            |
| TestSubview             | 1523* (13) |            |            |            |            |            |            |            |
| TEStyleHandle           | 285 (38)   | 321 (38)   | 578 (40)   | 1037 (40)  | 1485 (40)  | 1487 (40)  | 1520 (40)  |            |
| TEStyleInsert           | 141 (41)   | 230 (41)   |            |            |            |            |            |            |
| TEStyleNew              | 1225 (40)  |            |            |            |            |            |            |            |
| TESysJust               | 3178* ( 6) | 3194 ( 6)  |            |            |            |            |            |            |
| tesysJust               | 394* (40)  |            |            |            |            |            |            |            |
| TEUpdate                | 3209 ( 6)  | 1009 (40)  | 1029 (40)  |            |            |            |            |            |
| TEViewTemplate          | 69 (38)    | 70* (38)   | 117 (40)   | 137 (40)   |            |            |            |            |
| TEViewTemplatePtr       | 69* (38)   | 89 (40)    | 130 (40)   | 136 (40)   |            |            |            |            |
| TEvtHandler             | 583* (13)  | 585 (13)   | 593* (13)  | 593 (13)   | 594* (13)  | 597* (13)  | 601* (13)  | 602* (13)  |
|                         | 607* (13)  | 613* (13)  | 615* (13)  | 622* (13)  | 633* (13)  | 637* (13)  | 642* (13)  | 646* (13)  |
|                         | 650* (13)  | 652* (13)  | 656* (13)  | 658* (13)  | 682* (13)  | 794 (13)   | 799 (13)   | 800 (13)   |
|                         | 804 (13)   | 805 (13)   | 806 (13)   | 812 (13)   | 1077* (13) | 1459* (13) | 1909 (13)  | 2024 (13)  |
|                         | 2324* (13) | 2437 (13)  | 2679 (13)  | 2787 (13)  | 12* (15)   | 12 (15)    | 23* (15)   | 39* (15)   |
|                         | 57* (15)   | 105* (15)  | 201* (15)  | 213* (15)  | 224* (15)  | 233* (15)  | 245* (15)  | 262* (15)  |
|                         | 276* (15)  | 287* (15)  | 301* (15)  | 313* (15)  | 323* (15)  | 337* (15)  | 506 (16)   | 881 (16)   |
|                         | 882 (16)   | 887 (16)   | 888 (16)   | 1021 (16)  | 1022 (16)  | 1022 (16)  | 1028 (16)  | 1029 (16)  |
|                         | 1332 (16)  | 1337 (16)  | 1754 (16)  | 1834 (16)  | 1836 (16)  | 1837 (16)  | 2787 (16)  | 892 (19)   |
| text                    | 536* ( 5)  | 599* ( 5)  | 2846* ( 6) | 2848 ( 6)  | 3175* ( 6) | 3208 ( 6)  |            |            |
| TextBox                 | 1287 ( 6)  | 2350 ( 6)  | 2848 ( 6)  | 245 (23)   |            |            |            |            |
| TextFace                | 820 (26)   | 363 (37)   | 484 (40)   |            |            |            |            |            |
| TextFont                | 819 (26)   | 362 (37)   | 483 (40)   |            |            |            |            |            |
| TextGridViewTemplate    | 110* ( 9)  | 116 ( 9)   | 2259 (10)  | 2284 (10)  |            |            |            |            |
| TextGridViewTemplatePtr | 116* ( 9)  | 2241 (10)  | 2275 (10)  | 2283 (10)  |            |            |            |            |
| TextSize                | 821 (26)   | 364 (37)   | 485 (40)   |            |            |            |            |            |
| TextStyle               | 418 ( 9)   | 439 ( 9)   | 532 ( 9)   | 2211 (10)  | 2236 (10)  | 2374 (10)  | 2451 (10)  | 2070 (13)  |
|                         | 2118 (13)  | 2786 (13)  | 130 (21)   | 404 (21)   | 338 (25)   | 345 (25)   | 355 (25)   | 816 (26)   |
|                         | 841 (26)   | 1094 (26)  | 719 (37)   | 36 (38)    | 120 (38)   | 153 (38)   | 185 (38)   | 277 (38)   |
|                         | 307 (38)   | 506 (38)   | 507 (38)   | 511 (38)   | 517 (38)   | 102 (39)   | 17 (40)    | 73 (40)    |
|                         | 274 (40)   | 300 (40)   | 410 (40)   | 544 (40)   | 795 (40)   | 1360 (40)  | 1366 (40)  | 350 (41)   |
|                         | 519 (41)   | 541 (41)   | 703 (41)   | 712 (41)   | 742 (41)   |            |            |            |
| TextStyleFields         | 2412 (10)  | 516 (21)   | 355* (25)  | 1094* (26) | 1648 (40)  | 617 (41)   | 618 (41)   |            |
| TextWidth               | 503 (40)   | 508 (40)   | 510 (40)   |            |            |            |            |            |
| tgPtr                   | 2275* (10) | 2283* (10) | 2286 (10)  | 2293 (10)  |            |            |            |            |
| TGridSelectCommand      | 639* ( 9)  | 646* ( 9)  | 650* ( 9)  | 653* ( 9)  | 658* ( 9)  | 672* ( 9)  | 711* ( 9)  | 734* ( 9)  |
|                         | 756* ( 9)  | 2778* (10) | 2806* (10) | 2820* (10) | 2832* (10) |            |            |            |
| TGridView               | 262* ( 9)  | 299* ( 9)  | 309* (10)  | 1081* (10) | 1125* (10) | 1138* (10) |            |            |
| TGridView               | 130* ( 9)  | 171* ( 9)  | 193* ( 9)  | 199* ( 9)  | 202* ( 9)  | 206* ( 9)  | 217* ( 9)  | 220* ( 9)  |
|                         | 223* ( 9)  | 227* ( 9)  | 235* ( 9)  | 236* ( 9)  | 240* ( 9)  | 244* ( 9)  | 248* ( 9)  | 252* ( 9)  |
|                         | 255* ( 9)  | 265* ( 9)  | 269* ( 9)  | 270* ( 9)  | 274* ( 9)  | 279* ( 9)  | 280* ( 9)  | 284* ( 9)  |
|                         | 285* ( 9)  | 289* ( 9)  | 290* ( 9)  | 294* ( 9)  | 302* ( 9)  | 306* ( 9)  | 309* ( 9)  | 312* ( 9)  |
|                         | 315* ( 9)  | 316* ( 9)  | 319* ( 9)  | 325* ( 9)  | 326* ( 9)  | 331* ( 9)  | 332* ( 9)  | 337* ( 9)  |
|                         | 338* ( 9)  | 343* ( 9)  | 346* ( 9)  | 350* ( 9)  | 353* ( 9)  | 356* ( 9)  | 359* ( 9)  | 362* ( 9)  |
|                         | 363* ( 9)  | 366* ( 9)  | 370* ( 9)  | 374* ( 9)  | 380* ( 9)  | 385* ( 9)  | 388* ( 9)  | 396* ( 9)  |
|                         | 412* ( 9)  | 640 ( 9)   | 647 ( 9)   | 677 ( 9)   | 681 ( 9)   | 714 ( 9)   | 737 ( 9)   | 760 ( 9)   |
|                         | 29* (10)   | 140* (10)  | 240* (10)  | 277* (10)  | 290* (10)  | 329* (10)  | 339* (10)  | 351* (10)  |
|                         | 367* (10)  | 381* (10)  | 395* (10)  | 432* (10)  | 488* (10)  | 542* (10)  | 563* (10)  | 619* (10)  |
|                         | 686* (10)  | 767* (10)  | 823* (10)  | 839* (10)  | 936* (10)  | 1033* (10) | 1045* (10) | 1057* (10) |
|                         | 1068* (10) | 1163* (10) | 1247* (10) | 1330* (10) | 1342* (10) | 1373* (10) | 1404* (10) | 1420* (10) |
|                         | 1448* (10) | 1515* (10) | 1615* (10) | 1715* (10) | 1727* (10) | 1739* (10) | 1751* (10) | 1762* (10) |
|                         | 1773* (10) | 1786* (10) | 1841* (10) | 1852* (10) | 1882* (10) | 1908* (10) | 1933* (10) | 1952* (10) |
|                         | 1984* (10) | 2044* (10) | 2058* (10) | 2072* (10) | 2138* (10) | 2779 (10)  | 2853 (10)  | 2869 (10)  |
|                         | 3079 (10)  | 3112 (10)  | 3146 (10)  |            |            |            |            |            |
| theAlphaLock            | 507* (13)  | 1401* (16) | 2549 (16)  |            |            |            |            |            |
| theArea                 | 2093* (13) | 260* (21)  | 264 (21)   | 267 (21)   |            |            |            |            |
| theAscent               | 269* (40)  | 707* (41)  | 738 (41)   | 765 (41)   |            |            |            |            |
| theAutoKey              | 510* (13)  | 673 (16)   | 1404* (16) |            |            |            |            |            |
| theBits                 | 332* (25)  | 764* (26)  | 769 (26)   | 772 (26)   |            |            |            |            |
| theBlock                | 72* (35)   | 77* (35)   | 79* (35)   | 81* (35)   | 83 (35)    | 88 (35)    |            |            |
| theBottom               | 380* ( 9)  | 2044* (10) | 2047 (10)  |            |            |            |            |            |
| theBtnState             | 503* (13)  | 1398* (16) | 2555 (16)  |            |            |            |            |            |
| theCCursor              | 62* ( 4)   | 285* ( 4)  | 298 ( 4)   |            |            |            |            |            |
| theCell                 | 366* ( 9)  | 603* ( 9)  | 1933* (10) | 1936 (10)  | 1936 (10)  | 1939 (10)  | 1939 (10)  | 1939 (10)  |
|                         | 1939 (10)  | 2709* (10) | 2712 (10)  |            |            |            |            |            |
| theChars                | 3154* ( 6) | 3162* ( 6) | 3163 ( 6)  | 3167 ( 6)  |            |            |            |            |
| theClickCount           | 512* (13)  | 1405* (16) | 1566* (16) |            |            |            |            |            |
| theCmd                  | 435* (30)  | 455 (30)   | 456 (30)   |            |            |            |            |            |
| theCmdKey               | 647* ( 9)  | 678* ( 9)  | 715* ( 9)  | 738* ( 9)  | 761* ( 9)  | 643 (10)   | 652 (10)   | 661 (10)   |
|                         | 670 (10)   | 2779* (10) | 2790 (10)  | 2854* (10) | 2856 (10)  | 3080* (10) | 3082 (10)  | 3113* (10) |
|                         | 3115 (10)  | 3147* (10) | 3149 (10)  | 505* (13)  | 1399* (16) | 1536 (16)  | 2553 (16)  |            |
| theCmdNumber            | 12* (30)   | 76 (30)    | 124 (30)   | 130 (30)   | 331 (30)   | 336 (30)   |            |            |
| theColor                | 2116* (13) | 393* (21)  | 395 (21)   | 346* (25)  | 842* (26)  | 849 (26)   |            |            |
| theColorRsrc            | 21* (16)   | 38* (16)   | 39 (16)    | 41 (16)    | 42 (16)    | 44 (16)    | 45 (16)    | 364* (30)  |

|                    |           |           |           |           |           |           |           |           |  |
|--------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| theColorTab        | 373-(30)  | 374 (30)  | 376 (30)  | 377 (30)  | 379 (30)  | 380 (30)  |           |           |  |
| theCommand         | 20*(16)   | 25-(16)   | 27 (16)   | 29 (16)   | 30 (16)   | 30 (16)   | 31 (16)   | 32 (16)   |  |
|                    | 742*(13)  | 1623*(13) | 1990*(13) | 3098*(16) | 3244 (16) | 1300*(18) | 1316 (18) | 1323-(18) |  |
|                    | 1333-(18) | 687*(19)  | 694-(19)  | 709 (19)  |           |           |           |           |  |
| theControl         | 53*( 6)   | 57 ( 6)   | 59 ( 6)   | 61 ( 6)   | 2532*(13) | 36*(21)   | 38 (21)   | 40 (21)   |  |
|                    | 43 (21)   |           |           |           |           |           |           |           |  |
| theControlKey      | 509*(13)  | 1403-(16) | 2545 (16) |           |           |           |           |           |  |
| theCursor          | 61*( 4)   | 312*( 4)  | 322 ( 4)  | 325 ( 4)  |           |           |           |           |  |
| theData            | 245*(25)  | 350*(25)  | 186*(26)  | 859*(26)  | 935 (26)  |           |           |           |  |
| theEditText        | 86*( 5)   | 98*( 5)   | 100*( 5)  | 679*( 5)  | 408*( 6)  | 412 ( 6)  | 509*( 6)  | 515 ( 6)  |  |
|                    | 520 ( 6)  | 521 ( 6)  | 522 ( 6)  | 531 ( 6)  | 538*( 6)  | 721*( 6)  | 725 ( 6)  | 726 ( 6)  |  |
|                    | 727 ( 6)  | 730 ( 6)  | 732 ( 6)  | 2576*( 6) | 2582 ( 6) | 2586 ( 6) | 2588 ( 6) | 2589 ( 6) |  |
|                    | 2591 ( 6) | 2593 ( 6) | 2597 ( 6) | 2605 ( 6) | 2606 ( 6) | 2607 ( 6) | 2608 ( 6) |           |  |
| theElements        | 285*(38)  | 321*(38)  | 1037*(40) | 1041-(40) | 1485*(40) | 1505 (40) |           |           |  |
| theEnd             | 306*(38)  | 605*(38)  | 163*(39)  | 168 (39)  | 1360*(40) | 1378 (40) |           |           |  |
| theEntry           | 71*( 1)   | 77*( 1)   | 114*( 2)  | 117 ( 2)  | 129*( 2)  | 144*( 2)  | 147 ( 2)  | 149 ( 2)  |  |
|                    | 167*( 2)  | 170 ( 2)  | 181*( 2)  | 209*( 2)  | 212 ( 2)  | 239*( 2)  | 244*( 2)  | 247 ( 2)  |  |
|                    | 251-( 2)  | 252 ( 2)  | 259 ( 2)  | 270*( 2)  | 287-( 2)  | 288 ( 2)  | 289 ( 2)  | 298*( 2)  |  |
|                    | 303*( 2)  | 306 ( 2)  | 310-( 2)  | 311 ( 2)  | 312 ( 2)  | 321*( 2)  | 326*( 2)  | 329 ( 2)  |  |
|                    | 331 ( 2)  | 338-( 2)  | 339 ( 2)  | 346 ( 2)  |           |           |           |           |  |
| theEvent           | 716*(13)  | 1379*(16) | 1395 (16) | 1397 (16) | 2323*(16) | 2355 (16) | 2362 (16) |           |  |
| theEventInfo       | 708*(13)  | 714*(13)  | 721*(13)  | 723*(13)  | 728*(13)  | 732*(13)  | 737*(13)  | 744*(13)  |  |
|                    | 748*(13)  | 754*(13)  | 784*(13)  | 627*(16)  | 632 (16)  | 637 (16)  | 640 (16)  | 643 (16)  |  |
|                    | 646 (16)  | 649 (16)  | 652 (16)  | 656 (16)  | 659 (16)  | 1303*(16) | 1309 (16) | 1328*(16) |  |
|                    | 1339 (16) | 1352*(16) | 1361 (16) | 1384*(16) | 1391 (16) | 1395 (16) | 1409 (16) | 1411 (16) |  |
|                    | 1416 (16) | 1422 (16) | 1524*(16) | 1531 (16) | 1537 (16) | 1539 (16) | 1547*(16) | 1563 (16) |  |
|                    | 1596 (16) | 1614 (16) | 1644*(16) | 1648 (16) | 1656*(16) | 1668 (16) | 1703*(16) | 1709 (16) |  |
|                    | 2390*(16) | 2532*(16) | 2540 (16) |           |           |           |           |           |  |
| theFlag            | 1027*(19) |           |           |           |           |           |           |           |  |
| theFont            | 2274*(10) | 2281 (10) | 2284 (10) | 2293 (10) | 165*(21)  | 172 (21)  | 175 (21)  | 193 (21)  |  |
|                    | 345*(25)  | 841*(26)  | 846 (26)  | 129*(40)  | 134 (40)  | 137 (40)  | 156 (40)  |           |  |
| theFontInfo        | 2353*(10) | 2357 (10) | 2358 (10) | 301*(40)  | 323 (40)  | 325 (40)  | 416*(40)  | 447 (40)  |  |
|                    | 450 (40)  |           |           |           |           |           |           |           |  |
| theFrame           | 1252*( 6) | 1266 ( 6) | 1267 ( 6) | 1268 ( 6) | 1270 ( 6) | 1272 ( 6) | 1285 ( 6) | 1288 ( 6) |  |
| theHeight          | 268*(40)  | 299*(40)  | 312-(40)  | 314-(40)  | 326-(40)  | 326 (40)  | 331-(40)  | 332-(40)  |  |
|                    | 332 (40)  | 335 (40)  | 338 (40)  | 1577*(40) | 1580-(40) | 1581 (40) | 1584 (40) | 706*(41)  |  |
|                    | 738 (41)  | 764 (41)  |           |           |           |           |           |           |  |
| theIcon            | 327*( 5)  | 1577*( 6) | 1580 ( 6) |           |           |           |           |           |  |
| theID              | 2029*(16) | 2097 (16) | 2102 (16) | 488*(25)  | 490*(25)  | 1316*(26) | 1321 (26) | 1322 (26) |  |
|                    | 1330*(26) | 1332 (26) | 106*(28)  | 116 (28)  | 121 (28)  |           |           |           |  |
| theInk             | 84*(36)   | 372 (37)  | 376 (37)  | 553-(37)  | 565 (37)  | 565 (37)  | 881 (37)  | 916 (37)  |  |
|                    | 917 (37)  | 939 (37)  | 940 (37)  | 1055 (37) | 1061 (37) |           |           |           |  |
| theInt             | 492*(25)  | 494*(25)  | 1341*(26) | 1345 (26) | 1353*(26) | 1355 (26) |           |           |  |
| theInterior        | 86*(36)   | 382 (37)  | 883 (37)  | 1061-(37) | 1068-(37) | 1069 (37) | 1070 (37) | 1634 (37) |  |
|                    | 1644 (37) | 1870 (37) | 1886 (37) | 1896 (37) | 1915 (37) |           |           |           |  |
| theItem            | 733*(12)  | 755-(12)  | 756 (12)  | 759 (12)  | 1374*(37) | 1398 (37) | 1399 (37) | 1401 (37) |  |
|                    | 1404 (37) |           |           |           |           |           |           |           |  |
| theItemList        | 274*(14)  |           |           |           |           |           |           |           |  |
| theItemNumber      | 1957*(16) | 1973-(16) | 1979 (16) | 1998 (16) | 14*(30)   | 74 (30)   | 127 (30)  |           |  |
| theJust            | 541*( 5)  | 610*( 5)  | 2866*( 6) | 2868 ( 6) | 3256*( 6) | 3259 ( 6) | 3260 ( 6) |           |  |
| theLabel           | 275*( 5)  | 281*( 5)  | 822*( 6)  | 828 ( 6)  | 831 ( 6)  | 834 ( 6)  | 904*( 6)  | 910 ( 6)  |  |
|                    | 913 ( 6)  | 917 ( 6)  | 1031*( 6) | 1037 ( 6) | 1040 ( 6) | 1044 ( 6) | 1181*( 6) | 1187 ( 6) |  |
|                    | 1190 ( 6) | 1193 ( 6) | 1301*( 6) | 1304-( 6) | 1306-( 6) | 1348*( 6) | 1352 ( 6) | 1354 ( 6) |  |
|                    | 2342*( 6) | 2349-( 6) | 2350 ( 6) | 2350 ( 6) | 2372*( 6) | 2376 ( 6) | 2382 ( 6) | 2385 ( 6) |  |
|                    | 2389 ( 6) | 2391 ( 6) | 2394-( 6) | 2395-( 6) | 2403 ( 6) | 490*(25)  | 494*(25)  | 498*(25)  |  |
|                    | 1330*(26) | 1332 (26) | 1353*(26) | 1355 (26) | 1376*(26) | 1378 (26) |           |           |  |
| theLeft            | 380*( 9)  | 2044*(10) | 2047 (10) |           |           |           |           |           |  |
| theLength          | 273*(40)  |           |           |           |           |           |           |           |  |
| theLines           | 272*(40)  |           |           |           |           |           |           |           |  |
| theLongint         | 496*(25)  | 498*(25)  | 1364*(26) | 1368 (26) | 1376*(26) | 1378 (26) |           |           |  |
| theMargins         | 85*(36)   | 281-(37)  | 487 (37)  | 488 (37)  | 555 (37)  | 555 (37)  | 556 (37)  | 557 (37)  |  |
|                    | 558 (37)  | 882 (37)  | 1055-(37) | 1056 (37) | 1057 (37) | 1065-(37) | 1069 (37) | 1070 (37) |  |
|                    | 1098 (37) | 1835 (37) | 1890 (37) |           |           |           |           |           |  |
| theMenu            | 486*( 5)  | 2484*( 6) | 2500 ( 6) | 2505 ( 6) | 2510 ( 6) | 2512 ( 6) | 2514 ( 6) | 231*(29)  |  |
|                    | 355*(30)  | 366 (30)  | 373 (30)  |           |           |           |           |           |  |
| theMenuNumber      | 1956*(16) | 1972-(16) | 1976 (16) | 1979 (16) | 1996 (16) | 13*(30)   | 74 (30)   | 126 (30)  |  |
| TheMenuSetterUpper | 270*(29)  | 417*(30)  | 499 (30)  |           |           |           |           |           |  |
| theMode            | 306*(38)  | 298*(40)  | 318-(40)  | 319 (40)  | 413*(40)  | 442-(40)  | 443 (40)  | 1360*(40) |  |
|                    | 1379 (40) | 1384 (40) | 1389 (40) | 1395 (40) | 1397 (40) | 1399 (40) | 1401 (40) |           |  |
| theMouse           | 469*( 5)  | 588*( 5)  | 2239*( 6) | 3061*( 6) | 3074 ( 6) | 223*( 9)  | 619*(10)  | 1619*(13) |  |
|                    | 1621*(13) | 1625*(13) | 1630*(13) | 1987*(13) | 2091*(13) | 2099*(13) | 2175*(13) | 2248*(13) |  |
|                    | 2304*(13) | 1556*(16) | 1610-(16) | 1611 (16) | 1612 (16) | 3106*(16) | 3207 (16) | 3209 (16) |  |
|                    | 3232 (16) | 3252 (16) | 3255 (16) | 3256 (16) | 3257 (16) | 3261 (16) | 3262 (16) | 3266 (16) |  |
|                    | 3276 (16) | 3278 (16) | 3282 (16) | 3300 (16) | 3303 (16) | 3304 (16) | 3311 (16) | 3324 (16) |  |
|                    | 3332-(16) | 3342 (16) | 3345 (16) | 681*(18)  | 688 (18)  | 802*(18)  | 1263*(18) | 1274 (18) |  |
|                    | 1287 (18) | 1297*(18) | 1312 (18) | 1333 (18) | 684*(19)  | 709 (19)  | 248*(21)  | 253 (21)  |  |
|                    | 302*(21)  | 676*(21)  | 680 (21)  | 681 (21)  | 1048*(21) | 1055 (21) | 1059 (21) | 1062 (21) |  |
|                    | 1341*(21) | 1357 (21) | 1359 (21) | 1377 (21) | 195*(38)  | 883*(40)  | 891 (40)  |           |  |
| themouse           | 633 (10)  |           |           |           |           |           |           |           |  |
| theName            | 2031*(16) | 2097 (16) | 108*(28)  | 116 (28)  |           |           |           |           |  |
| theNumber          | 306*(25)  | 315*(25)  | 533*(26)  | 539 (26)  | 549*(26)  | 555 (26)  | 559 (26)  |           |  |
| theObject          | 344*(12)  | 347 (12)  | 362*(12)  | 365 (12)  | 61*(14)   | 19*(33)   | 20*(33)   |           |  |
| theOptionKey       | 508*(13)  | 1402-(16) | 2547 (16) |           |           |           |           |           |  |
| theOrigin          | 933*(37)  | 940-(37)  | 947 (37)  | 947 (37)  | 953 (37)  | 953 (37)  |           |           |  |
| thePaper           | 83*(36)   | 368 (37)  | 486 (37)  | 486 (37)  | 539-(37)  | 541 (37)  | 564 (37)  | 564 (37)  |  |
|                    | 880 (37)  | 1056 (37) | 1057 (37) | 1068 (37) | 1097 (37) | 1890 (37) |           |           |  |
| thePatch           | 77*(34)   | 85*(34)   | 95*(34)   | 102*(34)  | 64*(35)   | 83 (35)   | 84 (35)   | 85 (35)   |  |
|                    | 86 (35)   | 87 (35)   | 98*(35)   | 109 (35)  | 130 (35)  | 143*(35)  | 151 (35)  | 159 (35)  |  |
|                    | 177*(35)  | 185 (35)  | 194 (35)  | 212*(35)  | 219 (35)  | 220 (35)  | 224 (35)  | 231 (35)  |  |
|                    | 234 (35)  |           |           |           |           |           |           |           |  |
| thePattern         | 373*( 5)  | 1797*( 6) | 1800 ( 6) |           |           |           |           |           |  |
| thePEvent          | 501*(13)  | 632 (16)  | 1309 (16) | 1339 (16) | 1361 (16) | 1397-(16) | 1409 (16) | 1531 (16) |  |
|                    | 1563 (16) | 1566 (16) | 1596 (16) | 1605 (16) | 1648 (16) | 1668 (16) | 1709 (16) | 2540 (16) |  |
| thePicture         | 418*( 5)  | 1989*( 6) | 1992 ( 6) |           |           |           |           |           |  |
| thePoint           | 1502*(13) | 1506*(13) | 1510*(13) | 1514*(13) | 1799*(13) | 1801*(13) | 1402*(18) | 1404 (18) |  |
|                    | 1411*(18) | 1420 (18) | 1699*(18) | 1701 (18) | 1813*(18) | 1822 (18) | 364*(20)  | 366 (20)  |  |
|                    | 367 (20)  | 661*(20)  | 663 (20)  | 664 (20)  |           |           |           |           |  |
| thePort            | 92 ( 6)   | 290 (14)  | 384 (14)  | 1145 (14) | 3064 (16) | 18 (18)   | 20 (18)   | 22 (18)   |  |
|                    | 148 (18)  | 148 (18)  | 164 (18)  | 588 (18)  | 898 (18)  | 908 (18)  | 923 (18)  | 1025 (18) |  |

|               |           |           |           |           |           |           |           |           |
|---------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|               | 1045 (18) | 1046 (18) | 1160 (18) | 478 (19)  | 522 (19)  | 1010 (19) | 517 (20)  | 518 (20)  |
| thePt         | 520 (20)  | 700 (21)  | 313 (26)  | 86 (39)   | 88 (39)   | 89 (39)   |           |           |
|               | 310*(25)  | 674*(26)  | 677 (26)  | 677 (26)  | 678 (26)  | 678 (26)  | 679 (26)  | 679 (26)  |
| theQDMouse    | 680 (26)  | 680 (26)  | 682 (26)  | 31*(42)   |           |           |           |           |
|               | 3105*(16) | 3248*(16) | 3251 (16) | 3252 (16) | 3255 (16) | 3274 (16) | 3276 (16) | 3278 (16) |
| theQDPoint    | 3338*(16) | 3341 (16) | 3342 (16) | 3345 (16) |           |           |           |           |
| theRect       | 319*( 9)  | 1448*(10) | 1456 (10) |           |           |           |           |           |
|               | 465*( 5)  | 467*( 5)  | 1534*( 6) | 1540 ( 6) | 1549 ( 6) | 1553 ( 6) | 1753*( 6) | 1759 ( 6) |
|               | 1769 ( 6) | 1773 ( 6) | 1953*( 6) | 1958 ( 6) | 1968 ( 6) | 2207*( 6) | 2210 ( 6) | 2211 ( 6) |
|               | 2212 ( 6) | 2223*( 6) | 2226 ( 6) | 2227 ( 6) | 2228 ( 6) | 2806*( 6) | 2814 ( 6) | 2820 ( 6) |
|               | 3116*( 6) | 3122 ( 6) | 3123 ( 6) | 310*(25)  | 674*(26)  | 677 (26)  | 677 (26)  | 678 (26)  |
| theRegion     | 678 (26)  | 679 (26)  | 679 (26)  | 680 (26)  | 680 (26)  | 46*(42)   | 48*(42)   |           |
| theResource   | 374*( 9)  | 1984*(10) | 1995 (10) | 2023 (10) | 2025 (10) | 2029 (10) |           |           |
|               | 77*( 5)   | 79*( 5)   | 151*( 5)  | 153*( 5)  | 181*( 5)  | 183*( 5)  | 221*( 5)  | 223*( 5)  |
|               | 264*( 5)  | 266*( 5)  | 316*( 5)  | 318*( 5)  | 362*( 5)  | 364*( 5)  | 407*( 5)  | 409*( 5)  |
|               | 456*( 5)  | 458*( 5)  | 523*( 5)  | 525*( 5)  | 577*( 5)  | 579*( 5)  | 653*( 5)  | 655*( 5)  |
|               | 345*( 6)  | 351 ( 6)  | 353 ( 6)  | 367*( 6)  | 370 ( 6)  | 819*( 6)  | 826 ( 6)  | 830 ( 6)  |
|               | 844*( 6)  | 847 ( 6)  | 901*( 6)  | 908 ( 6)  | 912 ( 6)  | 924*( 6)  | 927 ( 6)  | 1028*( 6) |
|               | 1035 ( 6) | 1039 ( 6) | 1051*( 6) | 1054 ( 6) | 1178*( 6) | 1185 ( 6) | 1189 ( 6) | 1200*( 6) |
|               | 1203 ( 6) | 1481*( 6) | 1487 ( 6) | 1489 ( 6) | 1509*( 6) | 1512 ( 6) | 1703*( 6) | 1709 ( 6) |
|               | 1711 ( 6) | 1728*( 6) | 1731 ( 6) | 1907*( 6) | 1913 ( 6) | 1915 ( 6) | 1928*( 6) | 1931 ( 6) |
|               | 2139*( 6) | 2145 ( 6) | 2147 ( 6) | 2165*( 6) | 2168 ( 6) | 2731*( 6) | 2738 ( 6) | 2742 ( 6) |
|               | 2757*( 6) | 2760 ( 6) | 2974*( 6) | 2980 ( 6) | 2982 ( 6) | 2995*( 6) | 2998 ( 6) | 3441*( 6) |
|               | 3447 ( 6) | 3449 ( 6) | 3464*( 6) | 3467 ( 6) | 199*( 9)  | 202*( 9)  | 457*( 9)  | 460*( 9)  |
|               | 542*( 9)  | 240*(10)  | 246 (10)  | 248 (10)  | 277*(10)  | 280 (10)  | 2270*(10) | 2278 (10) |
|               | 2283 (10) | 2302*(10) | 2305 (10) | 2472*(10) | 2475 (10) | 1484*(13) | 1486*(13) | 1780*(13) |
|               | 1782*(13) | 1940*(13) | 1942*(13) | 2082*(13) | 2084*(13) | 2161*(13) | 2163*(13) | 2241*(13) |
|               | 2243*(13) | 2295*(13) | 2297*(13) | 239*(18)  | 244 (18)  | 266 (18)  | 275*(18)  | 278 (18)  |
|               | 171*(19)  | 178 (19)  | 182 (19)  | 214*(19)  | 217 (19)  | 80*(20)   | 86 (20)   | 88 (20)   |
|               | 111*(20)  | 114 (20)  | 162*(21)  | 169 (21)  | 174 (21)  | 204*(21)  | 207 (21)  | 567*(21)  |
|               | 569 (21)  | 579*(21)  | 991*(21)  | 997 (21)  | 999 (21)  | 1017*(21) | 1020 (21) | 1273*(21) |
|               | 1275 (21) | 1285*(21) | 165*(38)  | 167*(38)  | 127*(40)  | 132 (40)  | 136 (40)  | 164*(40)  |
| theResType    | 167 (40)  |           |           |           |           |           |           |           |
| theRight      | 175*(23)  | 180 (23)  | 182 (23)  |           |           |           |           |           |
| theRoutine    | 380*( 9)  | 2044*(10) | 2047 (10) |           |           |           |           |           |
|               | 78*(34)   | 86*(34)   | 96*(34)   | 99*(35)   | 117 (35)  | 131 (35)  | 144*(35)  | 161 (35)  |
|               | 178*(35)  | 196 (35)  |           |           |           |           |           |           |
| theRsrcID     | 486*( 5)  | 2484*( 6) | 2502 ( 6) | 2513 ( 6) |           |           |           |           |
| theScrollBar  | 1828*(13) | 349*(20)  | 354 (20)  | 354 (20)  | 357 (20)  |           |           |           |
| theScroller   | 1301*(21) | 1304 (21) | 1305 (21) |           |           |           |           |           |
| theSetting    | 385*( 9)  | 2058*(10) | 2060 (10) |           |           |           |           |           |
| theShiftKey   | 599 ( 6)  | 647*( 9)  | 678*( 9)  | 715*( 9)  | 738*( 9)  | 761*( 9)  | 642 (10)  | 651 (10)  |
|               | 660 (10)  | 670 (10)  | 2779*(10) | 2789 (10) | 2854*(10) | 2856 (10) | 3080*(10) | 3082 (10) |
|               | 3113*(10) | 3115 (10) | 3147*(10) | 3149 (10) | 506*(13)  | 1400*(16) | 2551 (16) | 891 (40)  |
| theSig        | 1318*(26) | 1320*(26) | 1322 (26) | 1323 (26) |           |           |           |           |
| theSize       | 2273*(10) | 2280*(10) | 2281 (10) | 2289 (10) | 164*(21)  | 171*(21)  | 172 (21)  | 189 (21)  |
|               | 231*(25)  | 240*(25)  | 346*(25)  | 59*(26)   | 64 (26)   | 65 (26)   | 66*(26)   | 166*(26)  |
|               | 842*(26)  | 848 (26)  |           |           |           |           |           |           |
| theStart      | 306*(38)  | 605*(38)  | 163*(39)  | 167 (39)  | 1360*(40) | 1378 (40) |           |           |
| theString     | 3512*( 6) | 3519 ( 6) | 3520 ( 6) | 3524 ( 6) | 3525 ( 6) | 245*(25)  | 350*(25)  | 186*(26)  |
|               | 859*(26)  | 914 (26)  | 915*(26)  | 915 (26)  | 917*(26)  | 917 (26)  | 928 (26)  | 929*(26)  |
|               | 929 (26)  | 931*(26)  | 931 (26)  | 936*(26)  | 940*(26)  | 942 (26)  | 944 (26)  | 946 (26)  |
|               | 948 (26)  | 950 (26)  | 952 (26)  | 954 (26)  | 958 (26)  | 959*(26)  | 959 (26)  | 962*(26)  |
|               | 965*(26)  | 966*(26)  | 971 (26)  | 977*(26)  | 979 (26)  | 983 (26)  | 984*(26)  | 984 (26)  |
|               | 989 (26)  | 990*(26)  | 990 (26)  | 992*(26)  | 992 (26)  | 994*(26)  | 994 (26)  | 997 (26)  |
|               | 999 (26)  | 1001 (26) | 1004*(26) | 1006*(26) | 1010*(26) | 1014*(26) | 1014 (26) | 1020*(26) |
|               | 1022*(26) | 1025 (26) | 1027*(26) | 1027 (26) | 1029*(26) | 1029 (26) | 1033*(26) | 1041*(26) |
|               | 1041 (26) | 1044 (26) | 1048 (26) | 1049*(26) | 1049 (26) | 1054 (26) | 1055*(26) | 1055 (26) |
|               | 1057*(26) | 1057 (26) | 1059*(26) | 1059 (26) | 1063*(26) | 1065*(26) | 1068*(26) | 1082*(26) |
|               | 1082 (26) |           |           |           |           |           |           |           |
| theStyle      | 346*(25)  | 842*(26)  | 847 (26)  | 307*(38)  | 274*(40)  | 300*(40)  | 319 (40)  | 322 (40)  |
|               | 410*(40)  | 436 (40)  | 443 (40)  | 446 (40)  | 1360*(40) | 1379 (40) | 1385 (40) | 1391 (40) |
|               | 1396 (40) | 1398 (40) | 1400 (40) | 1402 (40) |           |           |           |           |
| theStyles     | 285*(38)  | 321*(38)  | 578*(40)  | 609*(40)  | 610 (40)  | 1037*(40) | 1040*(40) | 1041 (40) |
| theSubview    | 1485*(40) | 1501 (40) | 1503 (40) | 1504 (40) | 1505 (40) |           |           |           |
|               | 543*( 6)  | 546 ( 6)  | 547 ( 6)  | 549 ( 6)  | 1526*(13) | 1532*(13) | 1534*(13) | 1570*(13) |
|               | 1576*(13) | 1789*(13) | 1791*(13) | 1817*(13) | 311*(18)  | 313 (18)  | 335*(18)  | 340 (18)  |
|               | 350 (18)  | 354 (18)  | 356 (18)  | 357 (18)  | 358 (18)  | 362 (18)  | 364 (18)  | 366 (18)  |
|               | 367 (18)  | 538*(18)  | 540 (18)  | 606*(18)  | 608 (18)  | 883*(18)  | 888 (18)  | 895 (18)  |
|               | 899 (18)  | 937*(18)  | 955*(18)  | 958 (18)  | 959 (18)  | 961 (18)  | 988*(18)  | 1269*(18) |
|               | 1275 (18) | 1276 (18) | 1277 (18) | 1307*(18) | 1313 (18) | 1314 (18) | 1315 (18) | 1390*(18) |
|               | 1434*(18) | 1436 (18) | 1460*(18) | 1463 (18) | 1465 (18) | 1466 (18) | 1467 (18) | 1475*(18) |
|               | 1478 (18) | 1480 (18) | 1481 (18) | 1482 (18) | 1495*(18) | 1497 (18) | 1536*(18) | 1539 (18) |
|               | 1541 (18) | 1542 (18) | 1544 (18) | 1562*(18) | 1564 (18) | 1668*(18) | 1670 (18) | 1683*(18) |
|               | 1691*(18) | 137*(20)  | 140 (20)  | 141 (20)  | 386*(20)  | 389 (20)  | 390 (20)  | 651*(20)  |
|               | 654 (20)  |           |           |           |           |           |           |           |
| theSubview    | 1520*(13) | 1523*(13) | 1528*(13) | 1530*(13) |           |           |           |           |
| theSuperview  | 2749*(16) | 2754 (16) | 2761*(16) | 2763*(16) | 2764 (16) | 2765 (16) | 2766 (16) |           |
| theTarget     | 1024*(14) | 1035*(14) | 1036 (14) | 1037 (14) |           |           |           |           |
| theTEView     | 616*( 5)  | 3330*( 6) | 3338 ( 6) |           |           |           |           |           |
| theText       | 116*( 5)  | 480*( 5)  | 534*( 5)  | 543*( 5)  | 597*( 5)  | 614*( 5)  | 667*( 6)  | 679 ( 6)  |
|               | 681 ( 6)  | 685*( 6)  | 685 ( 6)  | 2428*( 6) | 2431 ( 6) | 2433*( 6) | 2579*( 6) | 2597 ( 6) |
|               | 2598 ( 6) | 2734*( 6) | 2740 ( 6) | 2743 ( 6) | 2749 ( 6) | 2833*( 6) | 2837*( 6) | 2839*( 6) |
|               | 2877*( 6) | 2882 ( 6) | 2885 ( 6) | 3151*( 6) | 3159 ( 6) | 3165*( 6) | 3167 ( 6) | 3281*( 6) |
|               | 3291 ( 6) | 3293 ( 6) | 3304 ( 6) | 2317*(10) | 2319 (10) | 2324 (10) | 2184*(13) | 741*(21)  |
|               | 743*(21)  | 312*(38)  | 325*(38)  | 565*(38)  | 568*(38)  | 1437*(40) | 1443 (40) | 1443 (40) |
|               | 1518*(40) | 1527 (40) | 1533 (40) | 1534 (40) | 701*(41)  | 720 (41)  | 730 (41)  | 733 (41)  |
|               | 788*(41)  | 794 (41)  | 815*(41)  | 834*(41)  | 847 (41)  | 850 (41)  |           |           |
| theTextHandle | 1439*(40) | 1443 (40) | 1444 (40) |           |           |           |           |           |
| theTextStyle  | 2118*(13) | 404*(21)  | 406 (21)  | 338*(25)  | 345*(25)  | 816*(26)  | 819 (26)  | 820 (26)  |
|               | 821 (26)  | 822 (26)  | 841*(26)  | 844 (26)  |           |           |           |           |
| theTitle      | 174*(19)  | 180 (19)  | 183 (19)  | 203 (19)  |           |           |           |           |
| theTop        | 380*( 9)  | 2044*(10) | 2047 (10) |           |           |           |           |           |
| theTotal      | 307*( 9)  | 310*( 9)  | 436*(10)  | 458*(10)  | 463 (10)  | 465*(10)  | 465 (10)  | 474 (10)  |
|               | 476 (10)  | 847*(10)  | 877 (10)  | 944*(10)  | 974 (10)  | 1164*(10) | 1174*(10) | 1182*(10) |
|               | 1193*(10) | 1200*(10) | 1207*(10) | 1216*(10) | 1216 (10) | 1226*(10) | 1226 (10) | 1236 (10) |
|               | 1248*(10) | 1258*(10) | 1266*(10) | 1277*(10) | 1284*(10) | 1291*(10) | 1300*(10) | 1300 (10) |
|               | 1310*(10) | 1310 (10) | 1320 (10) | 1345*(10) | 1361 (10) | 1376*(10) | 1392 (10) | 1523*(10) |
|               | 1550 (10) | 1623*(10) | 1650 (10) | 1790*(10) | 1812*(10) | 1817 (10) | 1819*(10) | 1819 (10) |

|               |            |            |            |            |            |            |            |            |  |
|---------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| theTrap       | 1827 (10)  | 1829 (10)  |            |            |            |            |            |            |  |
| theTrapNum    | 275* (25)  | 100* (26)  | 104 (26)   | 104 (26)   | 454* (26)  | 458 (26)   |            |            |  |
|               | 77* (34)   | 85* (34)   | 95* (34)   | 98* (35)   | 109 (35)   | 121 (35)   | 121 (35)   | 130 (35)   |  |
|               | 131 (35)   | 131 (35)   | 143* (35)  | 151 (35)   | 165 (35)   | 165 (35)   | 177* (35)  | 185 (35)   |  |
|               | 200 (35)   | 200 (35)   |            |            |            |            |            |            |  |
| theTTECommand | 615* (38)  | 100* (39)  | 109 (39)   | 114 (39)   | 115 (39)   | 120 (39)   | 124 (39)   | 127 (39)   |  |
|               | 133 (39)   | 137 (39)   | 141 (39)   | 144 (39)   | 150 (39)   | 154 (39)   |            |            |  |
| theType       | 333* (14)  | 2030* (16) | 2097 (16)  | 107* (28)  | 116 (28)   | 804* (28)  | 822 (28)   |            |  |
| theValue      | 3480* (6)  | 3484 (6)   | 3485 (6)   |            |            |            |            |            |  |
| theView       | 1350* (21) | 1353 (21)  | 1412* (21) | 1415 (21)  |            |            |            |            |  |
| theViewInfo   | 111* (15)  | 137- (15)  | 142 (15)   | 177 (15)   | 180 (15)   |            |            |            |  |
| theViews      | 1456* (13) | 137 (15)   | 55 (18)    |            |            |            |            |            |  |
| theVMouse     | 1557* (16) | 1612 (16)  | 1614 (16)  |            |            |            |            |            |  |
| theVPT        | 31* (42)   | 33* (42)   |            |            |            |            |            |            |  |
| theVRect      | 46* (42)   | 48* (42)   |            |            |            |            |            |            |  |
| theWindow     | 1023* (14) | 1027- (14) | 1028 (14)  | 1030 (14)  | 1031 (14)  | 1032 (14)  | 1033 (14)  | 1035 (14)  |  |
|               | 1035 (14)  | 1037 (14)  | 1040 (14)  | 258* (20)  | 272- (20)  | 273 (20)   | 274 (20)   |            |  |
| thisCol       | 1166* (10) | 1194- (10) | 1202- (10) | 1208- (10) | 1211 (10)  | 1213 (10)  | 1217- (10) | 1217 (10)  |  |
|               | 1227- (10) | 1227 (10)  | 1228 (10)  | 1230 (10)  |            |            |            |            |  |
| thisRow       | 1250* (10) | 1278- (10) | 1286- (10) | 1292- (10) | 1295 (10)  | 1297 (10)  | 1301- (10) | 1301 (10)  |  |
|               | 1311- (10) | 1311 (10)  | 1312 (10)  | 1314 (10)  |            |            |            |            |  |
| thisViewID    | 1438* (13) | 150 (15)   | 229 (18)   | 253- (18)  |            |            |            |            |  |
| THPrint       | 234 (37)   | 537 (37)   | 1145 (37)  | 1155 (37)  | 1189 (37)  | 1289 (37)  | 1298 (37)  | 1326 (37)  |  |
|               | 1415 (37)  | 1467 (37)  | 1488 (37)  | 1694 (37)  | 1716 (37)  | 1725 (37)  | 1794 (37)  | 1797 (37)  |  |
|               | 1954 (37)  | 1954 (37)  | 1978 (37)  | 2067 (37)  |            |            |            |            |  |
| THz           | 336 (14)   | 412 (28)   | 1062 (28)  |            |            |            |            |            |  |
| TickCount     | 572 (14)   | 213 (16)   | 1774 (16)  | 2965 (16)  | 596 (17)   | 792 (19)   | 534 (37)   |            |  |
| TIcon         | 299* (5)   | 306* (5)   | 312* (5)   | 316* (5)   | 318* (5)   | 321* (5)   | 323* (5)   | 325* (5)   |  |
|               | 327* (5)   | 330* (5)   | 21 (6)     | 1390* (6)  | 1433* (6)  | 1481* (6)  | 1509* (6)  | 1521* (6)  |  |
|               | 1531* (6)  | 1563* (6)  | 1577* (6)  | 1590* (6)  |            |            |            |            |  |
| time          | 582* (17)  | 595 (17)   | 596 (17)   |            |            |            |            |            |  |
| title         | 1883* (13) | 2883* (13) | 37* (14)   | 519 (14)   | 1242* (14) | 1248- (14) | 1251 (14)  | 1251 (14)  |  |
|               | 1252 (14)  | 120 (19)   | 123 (19)   | 203 (19)   | 908* (19)  | 913 (19)   | 915 (19)   | 916 (19)   |  |
| titleHandle   | 2601 (16)  |            |            |            |            |            |            |            |  |
| titleMce      | 151* (6)   | 179- (6)   | 199 (6)    | 200 (6)    | 215 (6)    | 220 (6)    |            |            |  |
| TList         | 49* (11)   | 68* (11)   | 73* (11)   | 78* (11)   | 81* (11)   | 91* (11)   | 95* (11)   | 104* (11)  |  |
|               | 115* (11)  | 132* (11)  | 154* (11)  | 179* (11)  | 184* (11)  | 194* (11)  | 202* (11)  | 212* (11)  |  |
|               | 217* (11)  | 221* (11)  | 226* (11)  | 230* (11)  | 234* (11)  | 240* (11)  | 246* (11)  | 247* (11)  |  |
|               | 258* (11)  | 303 (11)   | 35 (12)    | 39 (12)    | 56* (12)   | 76* (12)   | 102* (12)  | 151* (12)  |  |
|               | 183* (12)  | 219* (12)  | 237* (12)  | 298* (12)  | 313* (12)  | 342* (12)  | 360* (12)  | 378* (12)  |  |
|               | 406* (12)  | 418* (12)  | 472* (12)  | 483* (12)  | 495* (12)  | 510* (12)  | 540* (12)  | 550* (12)  |  |
|               | 586* (12)  | 620* (12)  | 651* (12)  | 1079 (13)  | 1081 (13)  | 1462 (13)  | 2281 (13)  | 2644 (13)  |  |
|               | 2674 (13)  |            |            |            |            |            |            |            |  |
| tmpHTE        | 3181* (6)  | 3200- (6)  | 3201 (6)   | 3203 (6)   | 3208 (6)   | 3209 (6)   | 3210 (6)   |            |  |
| tmpName       | 1293* (17) | 1303 (17)  | 1347 (17)  | 1349 (17)  | 1370 (17)  |            |            |            |  |
| tmpRgn        | 46* (10)   | 104- (10)  | 105 (10)   | 107- (10)  | 108 (10)   | 110- (10)  | 111 (10)   | 113- (10)  |  |
|               | 114 (10)   | 116- (10)  | 117 (10)   | 143* (10)  | 204- (10)  | 205 (10)   | 207- (10)  | 208 (10)   |  |
|               | 210- (10)  | 211 (10)   | 213- (10)  | 214 (10)   | 216- (10)  | 217 (10)   | 2783* (10) | 2794- (10) |  |
|               | 2795 (10)  | 2796- (10) | 2797 (10)  |            |            |            |            |            |  |
| TNumberText   | 638* (5)   | 644* (5)   | 649* (5)   | 653* (5)   | 655* (5)   | 658* (5)   | 660* (5)   | 662* (5)   |  |
|               | 665* (5)   | 26 (6)     | 3395* (6)  | 3415* (6)  | 3441* (6)  | 3464* (6)  | 3476* (6)  | 3493* (6)  |  |
|               | 3509* (6)  | 3540* (6)  |            |            |            |            |            |            |  |
| toAddr        | 220* (25)  | 50* (26)   | 52 (26)    |            |            |            |            |            |  |
| TObject       | 28* (1)    | 51 (1)     | 59* (1)    | 73 (2)     | 144 (2)    | 275 (2)    | 326 (2)    | 672 (6)    |  |
|               | 49* (11)   | 81 (11)    | 91 (11)    | 95 (11)    | 104 (11)   | 115 (11)   | 132 (11)   | 133 (11)   |  |
|               | 154 (11)   | 155 (11)   | 179 (11)   | 184 (11)   | 194 (11)   | 202 (11)   | 212 (11)   | 260 (11)   |  |
|               | 272 (11)   | 277 (11)   | 283 (11)   | 283 (11)   | 18 (12)    | 76 (12)    | 102 (12)   | 183 (12)   |  |
|               | 237 (12)   | 283 (12)   | 298 (12)   | 313 (12)   | 313 (12)   | 317 (12)   | 344 (12)   | 362 (12)   |  |
|               | 378 (12)   | 418 (12)   | 472 (12)   | 483 (12)   | 495 (12)   | 510 (12)   | 510 (12)   | 514 (12)   |  |
|               | 664 (12)   | 677 (12)   | 715 (12)   | 726 (12)   | 726 (12)   | 733 (12)   | 736 (12)   | 583* (13)  |  |
|               | 2430* (13) | 2770 (13)  | 2988 (13)  | 2991 (13)  | 61 (14)    | 1155 (14)  | 1171 (14)  | 62* (31)   |  |
|               | 64 (31)    | 69 (31)    | 94* (31)   | 96* (31)   | 106 (31)   | 122 (31)   | 19 (33)    | 20 (33)    |  |
|               | 111 (33)   | 157* (33)  | 212 (33)   | 216 (33)   | 226* (33)  | 238* (33)  | 250* (33)  | 264* (33)  |  |
|               | 137 (32)   | 195 (32)   | 199 (32)   | 219 (32)   | 274* (32)  | 289* (32)  | 303* (32)  |            |  |
| toClass       | 240* (11)  | 586* (12)  | 593 (12)   | 594 (12)   | 596 (12)   |            |            |            |  |
| Toggle        | 192* (5)   | 232* (5)   | 939 (6)    | 965* (6)   | 1067 (6)   | 1094* (6)  |            |            |  |
| ToggleIf      | 194* (5)   | 234* (5)   | 974* (6)   | 1103* (6)  |            |            |            |            |  |
| toHL          | 220* (9)   | 248* (9)   | 542* (10)  | 553 (10)   | 563* (10)  | 576 (10)   | 577- (10)  | 581 (10)   |  |
|               | 587 (10)   | 599 (10)   | 1594* (13) | 2982* (13) | 1205* (14) | 1212 (14)  | 1215 (14)  | 1221 (14)  |  |
|               | 776* (18)  |            |            |            |            |            |            |            |  |
| toList        | 264* (27)  | 269* (27)  | 273* (27)  | 101* (28)  | 124 (28)   | 136* (28)  | 140 (28)   | 935* (28)  |  |
|               | 939 (28)   |            |            |            |            |            |            |            |  |
| Tone          | 227 (8)    | 228 (8)    |            |            |            |            |            |            |  |
| ToolIntf      | 14* (1)    | 58* (3)    | 14* (5)    | 136* (7)   | 25* (9)    | 17* (11)   | 45* (13)   | 44* (25)   |  |
|               | 130* (27)  | 110* (29)  | 15* (31)   | 53* (34)   | 14* (36)   | 21* (38)   | 14* (42)   |            |  |
| ToolTrap      | 461 (26)   |            |            |            |            |            |            |            |  |
| top           | 1267 (6)   | 1270- (6)  | 2198 (6)   | 2269 (6)   | 2410 (6)   | 2592 (6)   | 473- (10)  | 516 (10)   |  |
|               | 519 (10)   | 750 (10)   | 752 (10)   | 755- (10)  | 778- (10)  | 778 (10)   | 790 (10)   | 792 (10)   |  |
|               | 811- (10)  | 868- (10)  | 965- (10)  | 1488 (10)  | 1827- (10) | 1867 (10)  | 1868 (10)  | 2323 (10)  |  |
|               | 2890- (10) | 2890 (10)  | 2892 (10)  | 2897- (10) | 526- (14)  | 1032 (14)  | 3045 (16)  | 3066 (16)  |  |
|               | 3066 (16)  | 500 (18)   | 507 (18)   | 509 (18)   | 513 (18)   | 1046 (18)  | 1150 (18)  | 365- (19)  |  |
|               | 366 (19)   | 480- (19)  | 570 (19)   | 571 (19)   | 589 (19)   | 597 (19)   | 598 (19)   | 598 (19)   |  |
|               | 599 (19)   | 600 (19)   | 616 (19)   | 617 (19)   | 805 (19)   | 883 (19)   | 970 (19)   | 977 (19)   |  |
|               | 1014 (19)  | 237 (20)   | 238 (20)   | 264 (21)   | 783 (21)   | 784 (21)   | 785 (21)   | 976 (21)   |  |
|               | 166* (25)  | 679 (26)   | 679 (26)   | 731 (26)   | 731 (26)   | 989 (26)   | 1054 (26)  | 544 (37)   |  |
|               | 556 (37)   | 919 (37)   | 1780 (37)  | 383 (40)   | 437- (40)  | 437 (40)   | 449- (40)  | 451 (40)   |  |
|               | 469- (40)  | 469 (40)   | 531- (40)  | 531 (40)   | 534 (40)   | 1065- (40) | 1065 (40)  | 1065 (40)  |  |
|               | 50* (42)   |            |            |            |            |            |            |            |  |
| TopLeft       | 1153 (10)  |            |            |            |            |            |            |            |  |
| topLeft       | 169 (6)    | 1333 (10)  | 1862 (10)  | 290 (14)   | 603- (14)  | 851 (14)   | 881 (14)   | 958 (14)   |  |
|               | 1355* (16) | 1364 (16)  | 3135 (16)  | 3135 (16)  | 923 (18)   | 1036 (18)  | 1036 (18)  | 1138- (18) |  |
|               | 1148- (18) | 1182- (18) | 1616- (18) | 64 (19)    | 139 (19)   | 314 (19)   | 321 (19)   | 358 (19)   |  |
|               | 512 (19)   | 637 (19)   | 876- (19)  | 954 (19)   | 166 (20)   | 168 (20)   | 195 (20)   | 444 (20)   |  |
|               | 446 (20)   | 124 (23)   | 259 (23)   | 260 (23)   | 167* (25)  | 471 (26)   | 1223 (26)  | 1295 (26)  |  |
|               | 463 (37)   | 486 (37)   | 487 (37)   | 734 (37)   | 850 (37)   | 940 (37)   | 948 (37)   | 948 (37)   |  |
|               | 980 (37)   | 994 (37)   | 1056 (37)  | 1056 (37)  | 1069 (37)  | 1069 (37)  | 1097 (37)  | 1098 (37)  |  |
|               | 1262 (37)  | 1855 (37)  | 1860 (37)  | 1868 (37)  | 1890 (37)  | 1890 (37)  | 1890 (37)  | 1891 (37)  |  |
|               | 1898- (37) | 1900 (37)  | 1915 (37)  | 203 (39)   | 435- (40)  | 594 (40)   | 605 (40)   | 629 (40)   |  |
|               | 942 (40)   | 1220- (40) | 1220 (40)  | 1268- (40) | 1268 (40)  | 70* (42)   |            |            |  |

[illegible]



|                   |            |            |            |            |            |            |            |            |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|
|                   | 1719 (18)  | 1723 (18)  | 1853 (18)  | 56 (19)    | 57 (19)    | 106 (19)   | 311 (19)   | 338 (19)   |
|                   | 560 (19)   | 696 (19)   | 806 (19)   | 858 (19)   | 951 (19)   | 1014 (19)  | 172 (20)   | 173 (20)   |
|                   | 409 (20)   | 46 (21)    | 110 (21)   | 342 (21)   | 475 (21)   | 494 (21)   | 614 (21)   | 1040 (21)  |
|                   | 1372 (21)  | 19 (22)    | 20 (22)    | 23 (22)    | 210 (23)   | 272 (23)   | 162 (24)   | 77 (25)    |
|                   | 117 (26)   | 145 (27)   | 203 (28)   | 230 (28)   | 241 (28)   | 305 (28)   | 424 (28)   | 428 (28)   |
|                   | 466 (28)   | 658 (28)   | 683 (28)   | 769 (28)   | 775 (28)   | 823 (28)   | 855 (28)   | 919 (28)   |
|                   | 1025 (28)  | 485 (30)   | 498 (30)   | 500 (30)   | 508 (30)   | 199 (33)   | 84 (37)    | 127 (37)   |
|                   | 192 (37)   | 201 (37)   | 201 (37)   | 201 (37)   | 202 (37)   | 233 (37)   | 416 (37)   | 468 (37)   |
|                   | 527 (37)   | 574 (37)   | 675 (37)   | 699 (37)   | 700 (37)   | 701 (37)   | 704 (37)   | 729 (37)   |
|                   | 855 (37)   | 1164 (37)  | 1293 (37)  | 1465 (37)  | 1515 (37)  | 1730 (37)  | 1733 (37)  | 1809 (37)  |
|                   | 1814 (37)  | 1928 (37)  | 1939 (37)  | 1983 (37)  | 2050 (37)  | 58 (38)    | 60 (38)    | 217 (39)   |
|                   | 228 (39)   | 38 (40)    | 228 (40)   | 254 (40)   | 586 (40)   | 709 (40)   | 783 (40)   | 867 (40)   |
|                   | 890 (40)   | 919 (40)   | 923 (40)   | 1116 (40)  | 1146 (40)  | 1194 (40)  | 1423 (40)  | 1430 (40)  |
|                   | 1477 (40)  | 151 (41)   | 240 (41)   | 323 (41)   | 324 (41)   | 395 (41)   | 399 (41)   | 456 (41)   |
|                   | 500 (41)   | 556 (41)   | 560 (41)   | 645 (41)   | 744 (41)   | 837 (41)   | 914 (41)   |            |
| trueBuzzItem      | 258* (29)  | 577* (30)  |            |            |            |            |            |            |
| trueBuzzItem      | 582 (30)   |            |            |            |            |            |            |            |
| tsColor           | 2290 (10)  | 996 (16)   | 190 (21)   | 395- (21)  | 822 (26)   | 849- (26)  | 1103 (26)  | 153 (40)   |
|                   | 1398- (40) | 1398 (40)  | 1418 (40)  |            |            |            |            |            |
| tScriptUtil       | 88* (26)   | 121 (26)   |            |            |            |            |            |            |
| TScrollBar        | 2221* (13) | 2231* (13) | 2237* (13) | 2241* (13) | 2243* (13) | 2246* (13) | 2248* (13) | 2251* (13) |
|                   | 2253* (13) | 2255* (13) | 2257* (13) | 2259* (13) | 2261* (13) | 2263* (13) | 2265* (13) | 2270* (13) |
|                   | 2279* (13) | 152 (20)   | 14 (21)    | 20 (21)    | 941* (21)  | 962* (21)  | 991* (21)  | 1017* (21) |
|                   | 1029* (21) | 1048* (21) | 1081* (21) | 1090* (21) | 1099* (21) | 1108* (21) | 1128* (21) | 1142* (21) |
|                   | 1157* (21) | 1169* (21) | 1180* (21) |            |            |            |            |            |
| TScroller         | 1548 (13)  | 1641 (13)  | 1756* (13) | 1769* (13) | 1776* (13) | 1780* (13) | 1782* (13) | 1785* (13) |
|                   | 1789* (13) | 1791* (13) | 1795* (13) | 1799* (13) | 1801* (13) | 1805* (13) | 1809* (13) | 1813* (13) |
|                   | 1815* (13) | 1817* (13) | 1819* (13) | 1821* (13) | 1826* (13) | 1828* (13) | 1831* (13) | 1833* (13) |
|                   | 1835* (13) | 1837* (13) | 1839* (13) | 1841* (13) | 1843* (13) | 1845* (13) | 1850* (13) | 1850 (13)  |
|                   | 1855* (13) | 2287 (13)  | 2458 (13)  | 2462 (13)  | 475 (14)   | 834 (14)   | 901 (14)   | 3103 (16)  |
|                   | 553 (18)   | 1190 (18)  | 1193 (18)  | 13* (20)   | 46* (20)   | 80* (20)   | 111* (20)  | 122* (20)  |
|                   | 137* (20)  | 148* (20)  | 182* (20)  | 206* (20)  | 254* (20)  | 294* (20)  | 317* (20)  | 328* (20)  |
|                   | 339* (20)  | 339 (20)   | 349* (20)  | 364* (20)  | 374* (20)  | 386* (20)  | 397* (20)  | 428* (20)  |
|                   | 457* (20)  | 479* (20)  | 528* (20)  | 551* (20)  | 592* (20)  | 602* (20)  | 637* (20)  | 651* (20)  |
|                   | 661* (20)  | 1208 (21)  | 1301 (21)  | 1350 (21)  | 1412 (21)  | 13 (22)    | 136 (38)   | 230 (38)   |
|                   | 186 (39)   | 240 (40)   |            |            |            |            |            |            |
| tsFace            | 2288 (10)  | 994 (16)   | 188 (21)   | 820 (26)   | 847- (26)  | 1101 (26)  | 151 (40)   | 1396- (40) |
|                   | 1396 (40)  | 1415 (40)  |            |            |            |            |            |            |
| tsFont            | 2281 (10)  | 993 (16)   | 172 (21)   | 819 (26)   | 846- (26)  | 1100 (26)  | 134 (40)   | 1186 (40)  |
|                   | 1391- (40) | 1391 (40)  | 1393 (40)  | 1414 (40)  |            |            |            |            |
| tSndDoCommand     | 90* (26)   | 123 (26)   |            |            |            |            |            |            |
| TSortedList       | 49* (1)    | 258* (11)  | 260* (11)  | 272* (11)  | 277* (11)  | 283* (11)  | 664* (12)  | 677* (12)  |
|                   | 715* (12)  | 726* (12)  |            |            |            |            |            |            |
| TSPtr             | 102* (39)  | 703* (41)  | 710 (41)   | 766 (41)   |            |            |            |            |
| TSScrollBar       | 1761 (13)  | 1828 (13)  | 2279* (13) | 2284* (13) | 2291* (13) | 2295* (13) | 2297* (13) | 2300* (13) |
|                   | 2302* (13) | 2304* (13) | 2307* (13) | 2309* (13) | 2314* (13) | 474 (14)   | 211 (20)   | 349 (20)   |
|                   | 1205* (21) | 1244* (21) | 1273* (21) | 1285* (21) | 1296* (21) | 1319* (21) | 1341* (21) | 1384* (21) |
|                   | 1404* (21) | 1430* (21) |            |            |            |            |            |            |
| tsSize            | 2280 (10)  | 995 (16)   | 171 (21)   | 821 (26)   | 848- (26)  | 1102 (26)  | 152 (40)   | 1400- (40) |
|                   | 1400 (40)  | 1400 (40)  | 1402- (40) | 1402 (40)  | 1413 (40)  |            |            |            |
| TStaticText       | 505* (5)   | 513* (5)   | 519* (5)   | 523* (5)   | 525* (5)   | 528* (5)   | 530* (5)   | 532* (5)   |
|                   | 534* (5)   | 536* (5)   | 539* (5)   | 541* (5)   | 543* (5)   | 546* (5)   | 561* (5)   | 24 (6)     |
|                   | 2668* (6)  | 2708* (6)  | 2731* (6)  | 2757* (6)  | 2769* (6)  | 2779* (6)  | 2803* (6)  | 2833* (6)  |
|                   | 2846* (6)  | 2855* (6)  | 2866* (6)  | 2877* (6)  | 2902* (6)  |            |            |            |
| TStdPrintHandler  | 92* (36)   | 192* (36)  | 201* (36)  | 204* (36)  | 211* (36)  | 214* (36)  | 218* (36)  | 231* (36)  |
|                   | 239* (36)  | 249* (36)  | 252* (36)  | 255* (36)  | 261* (36)  | 265* (36)  | 270* (36)  | 274* (36)  |
|                   | 279* (36)  | 294* (36)  | 300* (36)  | 306* (36)  | 312* (36)  | 315* (36)  | 319* (36)  | 324* (36)  |
|                   | 339* (36)  | 342* (36)  | 345* (36)  | 348* (36)  | 353* (36)  | 361* (36)  | 365* (36)  | 373* (36)  |
|                   | 377* (36)  | 380* (36)  | 383* (36)  | 386* (36)  | 390* (36)  | 393* (36)  | 401* (36)  | 404* (36)  |
|                   | 410* (36)  | 424* (36)  | 427* (36)  | 431* (36)  | 434* (36)  | 441* (36)  | 444* (36)  | 449* (36)  |
|                   | 452* (36)  | 457* (36)  | 462* (36)  | 466* (36)  | 483 (36)   | 492 (36)   | 176 (37)   | 245* (37)  |
|                   | 316* (37)  | 348* (37)  | 393* (37)  | 407* (37)  | 436* (37)  | 479* (37)  | 496* (37)  | 588* (37)  |
|                   | 604* (37)  | 624* (37)  | 654* (37)  | 693* (37)  | 712* (37)  | 769* (37)  | 829* (37)  | 839* (37)  |
|                   | 874* (37)  | 912* (37)  | 929* (37)  | 971* (37)  | 1009* (37) | 1035* (37) | 1046* (37) | 1081* (37) |
|                   | 1092* (37) | 1107* (37) | 1118* (37) | 1203* (37) | 1224* (37) | 1247* (37) | 1277* (37) | 1315* (37) |
|                   | 1369* (37) | 1427* (37) | 1559* (37) | 1569* (37) | 1617* (37) | 1678* (37) | 1772* (37) | 1790* (37) |
|                   | 1831* (37) | 1844* (37) | 1881* (37) | 1909* (37) | 1923* (37) | 1968* (37) | 1991* (37) | 2018* (37) |
|                   | 2031* (37) | 2048* (37) | 2058* (37) | 2092 (37)  |            |            |            |            |
| TTECommand        | 182 (38)   | 368* (38)  | 400* (38)  | 403* (38)  | 408* (38)  | 411* (38)  | 415* (38)  | 421* (38)  |
|                   | 427* (38)  | 430* (38)  | 433* (38)  | 438* (38)  | 442* (38)  | 448* (38)  | 459* (38)  | 488* (38)  |
|                   | 501* (38)  | 542* (38)  | 615 (38)   | 100 (39)   | 757 (40)   | 761 (40)   | 13* (41)   | 93* (41)   |
|                   | 109* (41)  | 121* (41)  | 161* (41)  | 188* (41)  | 203* (41)  | 213* (41)  | 249* (41)  | 263* (41)  |
|                   | 279* (41)  | 299* (41)  |            |            |            |            |            |            |
| TTECutCopyCommand | 459* (38)  | 465* (38)  | 469* (38)  | 474* (38)  | 478* (38)  | 759 (40)   | 831 (40)   | 320* (41)  |
|                   | 333* (41)  | 345* (41)  | 416* (41)  |            |            |            |            |            |
| TTEPasteCommand   | 488* (38)  | 492* (38)  | 760 (40)   | 832 (40)   | 431* (41)  |            |            |            |
| TTEStyleCommand   | 186 (38)   | 501* (38)  | 511* (38)  | 517* (38)  | 520* (38)  | 523* (38)  | 526* (38)  | 528* (38)  |
|                   | 532* (38)  | 796 (40)   | 798 (40)   | 519* (41)  | 541* (41)  | 551* (41)  | 567* (41)  | 582* (41)  |
|                   | 599* (41)  | 610* (41)  |            |            |            |            |            |            |
| TTETypingCommand  | 116 (38)   | 189 (38)   | 542* (38)  | 552* (38)  | 557* (38)  | 562* (38)  | 565* (38)  | 568* (38)  |
|                   | 571* (38)  | 574* (38)  | 577* (38)  | 580* (38)  | 586* (38)  | 697 (40)   | 812 (40)   | 814 (40)   |
|                   | 632* (41)  | 665* (41)  | 677* (41)  | 701* (41)  | 788* (41)  | 813* (41)  | 880* (41)  | 894* (41)  |
|                   | 909* (41)  | 937* (41)  |            |            |            |            |            |            |
| TTEView           | 565 (5)    | 673* (5)   | 86* (38)   | 141* (38)  | 160* (38)  | 165* (38)  | 167* (38)  | 170* (38)  |
|                   | 173* (38)  | 178* (38)  | 182* (38)  | 185* (38)  | 189* (38)  | 192* (38)  | 195* (38)  | 199* (38)  |
|                   | 202* (38)  | 205* (38)  | 211* (38)  | 214* (38)  | 218* (38)  | 221* (38)  | 227* (38)  | 230* (38)  |
|                   | 233* (38)  | 236* (38)  | 240* (38)  | 243* (38)  | 246* (38)  | 256* (38)  | 259* (38)  | 262* (38)  |
|                   | 267* (38)  | 270* (38)  | 273* (38)  | 276* (38)  | 285* (38)  | 290* (38)  | 294* (38)  | 297* (38)  |
|                   | 301* (38)  | 306* (38)  | 312* (38)  | 317* (38)  | 321* (38)  | 325* (38)  | 330* (38)  | 335* (38)  |
|                   | 340* (38)  | 343* (38)  | 347* (38)  | 353* (38)  | 356* (38)  | 372 (38)   | 400 (38)   | 465 (38)   |
|                   | 492 (38)   | 511 (38)   | 552 (38)   | 9 (39)     | 23 (39)    | 13* (40)   | 70* (40)   | 127* (40)  |
|                   | 164* (40)  | 176* (40)  | 201* (40)  | 214* (40)  | 240* (40)  | 265* (40)  | 295* (40)  | 347* (40)  |
|                   | 366* (40)  | 391* (40)  | 543* (40)  | 563* (40)  | 573* (40)  | 641* (40)  | 675* (40)  | 695* (40)  |
|                   | 757* (40)  | 795* (40)  | 812* (40)  | 828* (40)  | 883* (40)  | 901* (40)  | 911* (40)  | 934* (40)  |
|                   | 950* (40)  | 982* (40)  | 1037* (40) | 1049* (40) | 1059* (40) | 1076* (40) | 1178* (40) | 1202* (40) |
|                   | 1247* (40) | 1258* (40) | 1290* (40) | 1348* (40) | 1360* (40) | 1437* (40) | 1452* (40) | 1464* (40) |
|                   | 1485* (40) | 1518* (40) | 1553* (40) | 1574* (40) | 1598* (40) | 1620* (40) | 1633* (40) | 13 (41)    |
|                   | 320 (41)   | 347 (41)   | 431 (41)   | 519 (41)   | 632 (41)   |            |            |            |
| TTextGridView     | 412* (9)   | 427* (9)   | 451* (9)   | 457* (9)   | 460* (9)   | 469* (9)   | 479* (9)   | 482* (9)   |

|                      |           |           |           |           |           |           |           |           |  |
|----------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
|                      | 485*( 9)  | 488*( 9)  | 493*( 9)  | 509*( 9)  | 2199*(10) | 2234*(10) | 2270*(10) | 2302*(10) |  |
| TTextListView        | 2315*(10) | 2334*(10) | 2351*(10) | 2372*(10) | 2389*(10) | 2405*(10) |           |           |  |
|                      | 509*( 9)  | 522*( 9)  | 542*( 9)  | 552*( 9)  | 556*( 9)  | 562*( 9)  | 565*( 9)  | 569*( 9)  |  |
|                      | 570*( 9)  | 571*( 9)  | 575*( 9)  | 579*( 9)  | 582*( 9)  | 585*( 9)  | 588*( 9)  | 591*( 9)  |  |
|                      | 592*( 9)  | 593*( 9)  | 597*( 9)  | 600*( 9)  | 603*( 9)  | 606*( 9)  | 611*( 9)  | 616*( 9)  |  |
|                      | 621*( 9)  | 2441*(10) | 2472*(10) | 2485*(10) | 2496*(10) | 2507*(10) | 2520*(10) | 2533*(10) |  |
|                      | 2546*(10) | 2557*(10) | 2573*(10) | 2589*(10) | 2600*(10) | 2611*(10) | 2627*(10) | 2640*(10) |  |
|                      | 2653*(10) | 2666*(10) | 2677*(10) | 2693*(10) | 2709*(10) | 2721*(10) | 2737*(10) | 2749*(10) |  |
| ttPtr                | 2761*(10) |           |           |           |           |           |           |           |  |
| turnItOn             | 130*(40)  | 136*(40)  | 139*(40)  |           |           |           |           |           |  |
|                      | 654*( 9)  | 687*( 9)  | 2821*(10) | 2931*(10) | 2129*(13) | 2487*(13) | 2539*(13) | 3189*(16) |  |
| TVertexSelectCommand | 3195*(16) | 66*(21)   | 68*(21)   | 461*(21)  | 82*(22)   |           |           |           |  |
| TView                | 756*( 9)  | 760*( 9)  | 765*( 9)  | 628*(10)  | 3146*(10) | 3161*(10) |           |           |  |
|                      | 56*( 5)   | 66*( 5)   | 73*( 5)   | 102*( 5)  | 104*( 5)  | 106*( 5)  | 108*( 5)  | 141*( 5)  |  |
|                      | 147*( 5)  | 171*( 5)  | 177*( 5)  | 186*( 5)  | 211*( 5)  | 217*( 5)  | 226*( 5)  | 254*( 5)  |  |
|                      | 260*( 5)  | 273*( 5)  | 306*( 5)  | 312*( 5)  | 352*( 5)  | 358*( 5)  | 397*( 5)  | 403*( 5)  |  |
|                      | 446*( 5)  | 452*( 5)  | 513*( 5)  | 519*( 5)  | 568*( 5)  | 573*( 5)  | 644*( 5)  | 649*( 5)  |  |
|                      | 253*( 6)  | 298*( 6)  | 447*( 6)  | 543*( 6)  | 561*( 6)  | 570*( 6)  | 588*( 6)  | 701*( 6)  |  |
|                      | 780*( 6)  | 796*( 6)  | 861*( 6)  | 878*( 6)  | 936*( 6)  | 989*( 6)  | 1006*( 6) | 1063*( 6) |  |
|                      | 1119*( 6) | 1158*( 6) | 1222*( 6) | 1227*( 6) | 1326*( 6) | 1331*( 6) | 1390*( 6) | 1433*( 6) |  |
|                      | 1613*( 6) | 1656*( 6) | 1833*( 6) | 1870*( 6) | 2024*( 6) | 2080*( 6) | 2668*( 6) | 2708*( 6) |  |
|                      | 2925*( 6) | 2951*( 6) | 3395*( 6) | 3415*( 6) | 130*( 9)  | 173*( 9)  | 193*( 9)  | 429*( 9)  |  |
|                      | 451*( 9)  | 524*( 9)  | 31*(10)   | 140*(10)  | 2201*(10) | 2234*(10) | 2443*(10) | 650*(13)  |  |
|                      | 650*(13)  | 652*(13)  | 652*(13)  | 1023*(13) | 1044*(13) | 1052*(13) | 1409*(13) | 1415*(13) |  |
|                      | 1421*(13) | 1459*(13) | 1461*(13) | 1475*(13) | 1476*(13) | 1482*(13) | 1482*(13) | 1484*(13) |  |
|                      | 1486*(13) | 1488*(13) | 1490*(13) | 1494*(13) | 1496*(13) | 1498*(13) | 1500*(13) | 1502*(13) |  |
|                      | 1506*(13) | 1510*(13) | 1514*(13) | 1520*(13) | 1520*(13) | 1522*(13) | 1523*(13) | 1523*(13) |  |
|                      | 1525*(13) | 1526*(13) | 1526*(13) | 1528*(13) | 1528*(13) | 1530*(13) | 1530*(13) | 1532*(13) |  |
|                      | 1532*(13) | 1534*(13) | 1534*(13) | 1536*(13) | 1538*(13) | 1538*(13) | 1542*(13) | 1544*(13) |  |
|                      | 1546*(13) | 1548*(13) | 1550*(13) | 1552*(13) | 1556*(13) | 1556*(13) | 1560*(13) | 1562*(13) |  |
|                      | 1564*(13) | 1566*(13) | 1568*(13) | 1570*(13) | 1570*(13) | 1574*(13) | 1576*(13) | 1576*(13) |  |
|                      | 1578*(13) | 1582*(13) | 1584*(13) | 1586*(13) | 1590*(13) | 1592*(13) | 1594*(13) | 1596*(13) |  |
|                      | 1598*(13) | 1602*(13) | 1604*(13) | 1606*(13) | 1608*(13) | 1610*(13) | 1613*(13) | 1615*(13) |  |
|                      | 1619*(13) | 1621*(13) | 1625*(13) | 1630*(13) | 1630*(13) | 1632*(13) | 1637*(13) | 1637*(13) |  |
|                      | 1639*(13) | 1641*(13) | 1643*(13) | 1645*(13) | 1647*(13) | 1649*(13) | 1651*(13) | 1653*(13) |  |
|                      | 1655*(13) | 1657*(13) | 1659*(13) | 1663*(13) | 1665*(13) | 1667*(13) | 1671*(13) | 1675*(13) |  |
|                      | 1684*(13) | 1687*(13) | 1691*(13) | 1695*(13) | 1698*(13) | 1702*(13) | 1705*(13) | 1709*(13) |  |
|                      | 1713*(13) | 1717*(13) | 1720*(13) | 1724*(13) | 1731*(13) | 1735*(13) | 1756*(13) | 1769*(13) |  |
|                      | 1776*(13) | 1789*(13) | 1791*(13) | 1817*(13) | 1903*(13) | 1936*(13) | 2060*(13) | 2073*(13) |  |
|                      | 2078*(13) | 2153*(13) | 2159*(13) | 2231*(13) | 2237*(13) | 2284*(13) | 2291*(13) | 2326*(13) |  |
|                      | 2339*(13) | 2450*(13) | 2462*(13) | 2550*(13) | 2565*(13) | 2626*(13) | 2629*(13) | 2673*(13) |  |
|                      | 2763*(13) | 2853*(13) | 2854*(13) | 2861*(13) | 2994*(13) | 253*(14)  | 262*(14)  | 831*(14)  |  |
|                      | 831*(14)  | 899*(14)  | 1024*(14) | 57*(15)   | 57*(15)   | 60*(15)   | 61*(15)   | 75*(15)   |  |
|                      | 82*(15)   | 105*(15)  | 105*(15)  | 109*(15)  | 113*(15)  | 114*(15)  | 468*(16)  | 1935*(16) |  |
|                      | 2489*(16) | 2746*(16) | 2749*(16) | 2751*(16) | 2761*(16) | 2980*(16) | 3005*(16) | 3102*(16) |  |
|                      | 125*(17)  | 222*(17)  | 436*(17)  | 976*(17)  | 1432*(17) | 179*(18)  | 179*(18)  | 219*(18)  |  |
|                      | 219*(18)  | 239*(18)  | 275*(18)  | 286*(18)  | 306*(18)  | 311*(18)  | 335*(18)  | 335*(18)  |  |
|                      | 376*(18)  | 394*(18)  | 523*(18)  | 533*(18)  | 538*(18)  | 553*(18)  | 570*(18)  | 579*(18)  |  |
|                      | 601*(18)  | 606*(18)  | 619*(18)  | 669*(18)  | 681*(18)  | 695*(18)  | 707*(18)  | 717*(18)  |  |
|                      | 727*(18)  | 737*(18)  | 746*(18)  | 746*(18)  | 756*(18)  | 766*(18)  | 776*(18)  | 784*(18)  |  |
|                      | 802*(18)  | 812*(18)  | 822*(18)  | 832*(18)  | 841*(18)  | 851*(18)  | 863*(18)  | 871*(18)  |  |
|                      | 883*(18)  | 937*(18)  | 937*(18)  | 947*(18)  | 947*(18)  | 949*(18)  | 950*(18)  | 955*(18)  |  |
|                      | 988*(18)  | 988*(18)  | 988*(18)  | 991*(18)  | 1000*(18) | 1064*(18) | 1076*(18) | 1089*(18) |  |
|                      | 1103*(18) | 1124*(18) | 1124*(18) | 1136*(18) | 1146*(18) | 1157*(18) | 1171*(18) | 1180*(18) |  |
|                      | 1190*(18) | 1209*(18) | 1225*(18) | 1237*(18) | 1263*(18) | 1263*(18) | 1267*(18) | 1269*(18) |  |
|                      | 1297*(18) | 1302*(18) | 1307*(18) | 1342*(18) | 1355*(18) | 1372*(18) | 1381*(18) | 1390*(18) |  |
|                      | 1390*(18) | 1390*(18) | 1393*(18) | 1402*(18) | 1411*(18) | 1414*(18) | 1429*(18) | 1434*(18) |  |
|                      | 1460*(18) | 1460*(18) | 1463*(18) | 1475*(18) | 1475*(18) | 1478*(18) | 1490*(18) | 1495*(18) |  |
|                      | 1508*(18) | 1516*(18) | 1526*(18) | 1536*(18) | 1536*(18) | 1551*(18) | 1562*(18) | 1610*(18) |  |
|                      | 1625*(18) | 1639*(18) | 1648*(18) | 1663*(18) | 1668*(18) | 1683*(18) | 1683*(18) | 1691*(18) |  |
|                      | 1691*(18) | 1699*(18) | 1708*(18) | 1738*(18) | 1746*(18) | 1765*(18) | 1782*(18) | 1793*(18) |  |
|                      | 1803*(18) | 1813*(18) | 1816*(18) | 1831*(18) | 1902*(18) | 83*(19)   | 13*(20)   | 46*(20)   |  |
|                      | 137*(20)  | 386*(20)  | 651*(20)  | 97*(21)   | 127*(21)  | 532*(21)  | 552*(21)  | 941*(21)  |  |
|                      | 962*(21)  | 1205*(21) | 1244*(21) | 13*(22)   | 42*(23)   | 13*(24)   | 192*(36)  | 245*(37)  |  |
| tWaitNextEvent       | 509*(37)  | 86*(38)   | 145*(38)  | 160*(38)  | 13*(40)   | 70*(40)   |           |           |  |
| TWindow              | 91*(26)   | 124*(26)  |           |           |           |           |           |           |  |
|                      | 512*( 6)  | 646*( 6)  | 3335*( 6) | 897*(13)  | 902*(13)  | 904*(13)  | 910*(13)  | 927*(13)  |  |
|                      | 1035*(13) | 1412*(13) | 1418*(13) | 1426*(13) | 1643*(13) | 1903*(13) | 1931*(13) | 1936*(13) |  |
|                      | 1940*(13) | 1942*(13) | 1945*(13) | 1949*(13) | 1951*(13) | 1953*(13) | 1957*(13) | 1961*(13) |  |
|                      | 1963*(13) | 1965*(13) | 1967*(13) | 1969*(13) | 1973*(13) | 1975*(13) | 1979*(13) | 1981*(13) |  |
|                      | 1983*(13) | 1987*(13) | 1994*(13) | 1996*(13) | 2000*(13) | 2002*(13) | 2004*(13) | 2006*(13) |  |
|                      | 2008*(13) | 2010*(13) | 2012*(13) | 2014*(13) | 2016*(13) | 2018*(13) | 2020*(13) | 2020*(13) |  |
|                      | 2022*(13) | 2024*(13) | 2026*(13) | 2031*(13) | 2035*(13) | 2631*(13) | 2676*(13) | 2854*(13) |  |
|                      | 2861*(13) | 2865*(13) | 2871*(13) | 476*(14)  | 832*(14)  | 833*(14)  | 899*(14)  | 900*(14)  |  |
|                      | 975*(14)  | 977*(14)  | 1020*(14) | 1023*(14) | 1027*(14) | 261*(16)  | 542*(16)  | 618*(16)  |  |
|                      | 706*(16)  | 872*(16)  | 1060*(16) | 1306*(16) | 1551*(16) | 1663*(16) | 1706*(16) | 1921*(16) |  |
|                      | 2135*(16) | 2161*(16) | 2324*(16) | 3004*(16) | 3375*(16) | 3383*(16) | 146*(17)  | 188*(17)  |  |
|                      | 232*(17)  | 454*(17)  | 735*(17)  | 744*(17)  | 1385*(17) | 1449*(17) | 1105*(18) | 1225*(18) |  |
|                      | 1749*(18) | 15*(19)   | 83*(19)   | 171*(19)  | 214*(19)  | 226*(19)  | 253*(19)  | 298*(19)  |  |
|                      | 334*(19)  | 346*(19)  | 376*(19)  | 396*(19)  | 404*(19)  | 432*(19)  | 444*(19)  | 455*(19)  |  |
|                      | 469*(19)  | 502*(19)  | 535*(19)  | 546*(19)  | 626*(19)  | 649*(19)  | 662*(19)  | 673*(19)  |  |
|                      | 673*(19)  | 684*(19)  | 719*(19)  | 747*(19)  | 761*(19)  | 773*(19)  | 784*(19)  | 801*(19)  |  |
|                      | 816*(19)  | 848*(19)  | 866*(19)  | 892*(19)  | 905*(19)  | 925*(19)  | 941*(19)  | 1002*(19) |  |
|                      | 1023*(19) | 258*(20)  | 1011*(37) |           |           |           |           |           |  |
| txFace               | 484*(40)  | 1415*(40) |           |           |           |           |           |           |  |
| txFont               | 483*(40)  | 1414*(40) |           |           |           |           |           |           |  |
| txSize               | 485*(40)  | 1413*(40) |           |           |           |           |           |           |  |
| TXWord               | 1685*(37) | 1725*(37) |           |           |           |           |           |           |  |
| typeList             | 872*(13)  | 337*(16)  | 346*(16)  | 347*(16)  | 349*(16)  | 351*(16)  | 356*(16)  | 368*(16)  |  |
|                      | 419*(16)  | 424*(16)  | 425*(16)  | 427*(16)  | 428*(16)  | 437*(16)  | 438*(16)  | 439*(16)  |  |
|                      | 452*(16)  | 2928*(16) | 2938*(16) | 2939*(16) |           |           |           |           |  |
| TypeName             | 83*(31)   | 174*(33)  | 274*(32)  | 279*(32)  | 308*(32)  |           |           |           |  |
| typeName             | 2988*(13) | 2991*(13) | 1155*(14) | 1159*(14) | 1160*(14) | 1171*(14) | 1181*(14) | 1182*(14) |  |
| -U-                  |           |           |           |           |           |           |           |           |  |
| UAssociation         | 5*( 1)    | 17*( 5)   | 28*( 9)   | 57*(13)   | 23*(36)   | 24*(38)   |           |           |  |
| UBusyCursor          | 15*( 3)   | 51*(13)   |           |           |           |           |           |           |  |
| UDialog              | 5*( 5)    |           |           |           |           |           |           |           |  |
| UFailure             | 17*( 1)   | 62*( 3)   | 16*( 5)   | 16*( 7)   | 27*( 9)   | 20*(11)   | 52*(13)   | 133*(27)  |  |

|                   |           |           |           |           |           |           |           |           |
|-------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| UGridView         | 113*(29)  | 18*(31)   | 22*(36)   | 23*(38)   |           |           |           |           |
| UList             | 19*( 9)   |           |           |           |           |           |           |           |
| UList             | 19*( 1)   | 17*( 5)   | 28*( 9)   | 5*(11)    | 56*(13)   | 22*(36)   | 24*(38)   |           |
| UMacApp           | 17*( 1)   | 28*( 9)   | 15*(13)   | 23*(36)   | 24*(38)   |           |           |           |
| UMAUtil           | 16*( 1)   | 60*( 3)   | 16*( 5)   | 138*( 7)  | 27*( 9)   | 19*(11)   | 48*(13)   | 28*(25)   |
|                   | 132*(27)  | 112*(29)  | 17*(31)   | 55*(34)   | 22*(36)   | 23*(38)   | 16*(42)   |           |
| UMemory           | 17*( 5)   | 27*( 9)   | 21*(11)   | 53*(13)   | 19*(27)   | 114*(29)  | 19*(31)   | 22*(36)   |
|                   | 24*(38)   |           |           |           |           |           |           |           |
| UMenuSetup        | 17*( 5)   | 27*( 9)   | 54*(13)   | 17*(29)   | 22*(36)   | 24*(38)   |           |           |
| underline         | 1036 (26) |           |           |           |           |           |           |           |
| undoCmd           | 2849*(16) | 2867*(16) | 2871*(16) | 2881*(16) | 2887 (16) |           |           |           |
| UndoIt            | 2476*(13) | 805 (16)  | 109*(22)  | 499*(36)  | 2165*(37) | 415*(38)  | 526*(38)  | 577*(38)  |
|                   | 279*(41)  | 582*(41)  | 894*(41)  | 897 (41)  |           |           |           |           |
| undoName          | 2804*(16) | 2818 (16) | 2819 (16) | 2825 (16) | 2828 (16) |           |           |           |
| undoState         | 2848*(16) | 2866*(16) | 2870*(16) | 2878*(16) | 2880*(16) | 2887 (16) |           |           |
| UnimplementedTrap | 87*(26)   | 103 (26)  |           |           |           |           |           |           |
| UnionRgn          | 502 (10)  | 2023 (10) | 2977 (10) |           |           |           |           |           |
| UnionVRect        | 1866 (10) | 68*(42)   |           |           |           |           |           |           |
| UnitNtryCnt       | 2056*(16) | 2068 (16) |           |           |           |           |           |           |
| UnitTable         | 2060*(16) | 2061 (16) |           |           |           |           |           |           |
| UnitTablePtr      | 2061*(16) | 2070 (16) |           |           |           |           |           |           |
| UNIV              | 214 ( 7)  | 62 ( 8)   | 2927 (13) | 2931 (13) | 2931 (13) | 2935 (13) | 2935 (13) | 2942 (13) |
|                   | 2942 (13) | 2948 (13) | 2951 (13) | 31 (18)   | 41 (18)   | 64 (18)   | 64 (18)   | 75 (18)   |
|                   | 87 (18)   | 97 (18)   | 97 (18)   | 136 (18)  | 136 (18)  | 220 (25)  | 237 (25)  | 240 (25)  |
|                   | 455 (25)  | 473 (25)  | 50 (26)   | 156 (26)  | 166 (26)  | 1194 (26) | 1208 (26) |           |
| UnloadAllSegments | 302 (14)  | 348 (14)  | 370 (14)  | 1907 (16) | 2289 (16) | 2337 (16) | 2454 (16) | 2474 (16) |
|                   | 2478 (16) | 2685 (16) | 2694 (16) | 2704 (16) | 2960 (16) | 257*(27)  | 731 (28)  | 1107*(28) |
|                   | 220 (37)  |           |           |           |           |           |           |           |
| UnloadSeg         | 1137 (28) |           |           |           |           |           |           |           |
| UnpatchAll        | 100 (14)  | 105*(34)  | 256*(35)  |           |           |           |           |           |
| UnpatchTrap       | 186 ( 4)  | 187 ( 4)  | 188 ( 4)  | 192 ( 4)  | 193 ( 4)  | 93 (14)   | 102*(34)  | 212*(35)  |
|                   | 229 (35)  | 260 (35)  |           |           |           |           |           |           |
| UntitledName      | 1224*(13) | 2155 (16) | 1463*(17) |           |           |           |           |           |
| UObject           | 18*( 1)   | 17*( 5)   | 27*( 9)   | 22*(11)   | 55*(13)   | 6*(31)    | 22*(36)   | 24*(38)   |
| UPatch            | 61*( 3)   | 50*(13)   | 134*(27)  | 13*(34)   |           |           |           |           |
| Update            | 1598*(13) | 1719 (16) | 1746*(18) | 264 (19)  | 274 (20)  |           |           |           |
| UpdateAllWindows  | 774*(13)  | 447 (16)  | 3361*(16) | 926 (17)  | 1306 (37) |           |           |           |
| updateEvt         | 645 (16)  | 2594 (16) | 2596 (16) | 91 (37)   |           |           |           |           |
| updateMask        | 3367 (16) | 54 (37)   |           |           |           |           |           |           |
| updateRgn         | 273 (20)  |           |           |           |           |           |           |           |
| UprChar           | 366*(25)  | 1112*(26) | 1116*(26) | 1118*(26) |           |           |           |           |
| UPrinting         | 5*(36)    |           |           |           |           |           |           |           |
| UprString         | 259 (26)  |           |           |           |           |           |           |           |
| UprStrings        | 597 (12)  | 369*(25)  | 1127*(26) | 110 (32)  |           |           |           |           |
| upToSelWidth      | 406*(40)  | 499*(40)  | 504 (40)  | 508*(40)  | 516 (40)  | 521 (40)  |           |           |
| usedBlks          | 1048*(17) | 1149*(17) | 1152 (17) | 1184*(17) | 1185 (17) | 1189 (17) |           |           |
| UseResFile        | 866 (17)  |           |           |           |           |           |           |           |
| UserOMMap         | 112 ( 4)  | 372*(25)  | 1144*(26) | 120 (28)  | 122 (28)  | 919 (40)  |           |           |
| usesDataFork      | 1141*(13) | 11*(17)   | 47 (17)   |           |           |           |           |           |
| usesRsrcFork      | 1141*(13) | 11*(17)   | 48 (17)   |           |           |           |           |           |
| UseTempRgn        | 2837*(13) | 1266*(14) | 3056 (16) | 161 (18)  | 584 (18)  | 1588 (18) | 483 (19)  | 578 (19)  |
|                   | 507 (20)  | 998 (40)  | 1014 (40) |           |           |           |           |           |
| useWFont          | 642 (21)  |           |           |           |           |           |           |           |
| UTableBase        | 2057*(16) | 2070 (16) |           |           |           |           |           |           |
| UTableView        | 19*( 5)   | 5*(38)    |           |           |           |           |           |           |
| UTrace            | 141*( 7)  | 59*(13)   | 136*(27)  | 116*(29)  | 24*(31)   |           |           |           |
| UViewCoords       | 16*( 5)   | 27*( 9)   | 49*(13)   | 22*(36)   | 23*(38)   | 5*(42)    |           |           |
| UWriteLnWindow    | 26*(38)   |           |           |           |           |           |           |           |
| UWriteLnWindow    | 140*( 7)  | 62*(13)   | 48*(25)   | 21*(31)   |           |           |           |           |

-V-

|                     |           |           |           |           |           |           |           |           |
|---------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| v                   | 604*( 5)  | 1261 ( 6) | 1269 ( 6) | 2272 ( 6) | 2592 ( 6) | 2594 ( 6) | 3218*( 6) | 3221 ( 6) |
|                     | 3223 ( 6) | 3347*( 6) | 3347 ( 6) | 98*(10)   | 316*(10)  | 318*(10)  | 356*(10)  | 400 (10)  |
|                     | 401 (10)  | 406 (10)  | 414 (10)  | 496 (10)  | 497 (10)  | 635*(10)  | 747 (10)  | 747 (10)  |
|                     | 787 (10)  | 787 (10)  | 805*(10)  | 1092 (10) | 1094 (10) | 1112*(10) | 1460 (10) | 1463 (10) |
|                     | 1464 (10) | 1488 (10) | 1496 (10) | 1867*(10) | 1936 (10) | 1939 (10) | 1939 (10) | 2081 (10) |
|                     | 2082*(10) | 2110*(10) | 2112*(10) | 2112 (10) | 2117*(10) | 2118*(10) | 2118 (10) | 2119 (10) |
|                     | 2121*(10) | 2121 (10) | 2123*(10) | 2123 (10) | 2127*(10) | 2127 (10) | 2127 (10) | 2498 (10) |
|                     | 2576 (10) | 2629 (10) | 2683*(10) | 2712 (10) | 2726*(10) | 2890 (10) | 2892 (10) | 2897 (10) |
|                     | 2897 (10) | 2899 (10) | 2899 (10) | 2944 (10) | 2945 (10) | 2955 (10) | 460 (13)  | 1574*(13) |
|                     | 1813*(13) | 1841*(13) | 1961*(13) | 629 (14)  | 630 (14)  | 863 (14)  | 884 (14)  | 928*(14)  |
|                     | 928 (14)  | 961 (14)  | 1195 (14) | 268 (15)  | 268 (15)  | 966 (16)  | 2573 (16) | 3045*(16) |
|                     | 3045 (16) | 3074 (16) | 3074 (16) | 3164 (16) | 3164 (16) | 3285 (16) | 3285 (16) | 3310 (16) |
|                     | 3318 (16) | 197 (18)  | 256 (18)  | 383 (18)  | 383 (18)  | 385 (18)  | 451 (18)  | 472 (18)  |
|                     | 481 (18)  | 500 (18)  | 500 (18)  | 509 (18)  | 513 (18)  | 630 (18)  | 890 (18)  | 896 (18)  |
|                     | 1033 (18) | 1046 (18) | 1047 (18) | 1150 (18) | 1429*(18) | 1440 (18) | 1440 (18) | 1445*(18) |
|                     | 1445 (18) | 1529 (18) | 1568 (18) | 1570 (18) | 1573*(18) | 1630 (18) | 1717 (18) | 1729 (18) |
|                     | 1805 (18) | 1913 (18) | 110 (19)  | 110 (19)  | 110 (19)  | 325 (19)  | 365 (19)  | 365 (19)  |
|                     | 761*(19)  | 765 (19)  | 822 (19)  | 831 (19)  | 835 (19)  | 883 (19)  | 989*(19)  | 989 (19)  |
|                     | 991 (19)  | 22 (20)   | 33*(20)   | 53 (20)   | 61*(20)   | 94 (20)   | 95 (20)   | 97 (20)   |
|                     | 99 (20)   | 127 (20)  | 128 (20)  | 158 (20)  | 172 (20)  | 173 (20)  | 194 (20)  | 225 (20)  |
|                     | 225 (20)  | 237 (20)  | 238 (20)  | 242 (20)  | 242 (20)  | 245 (20)  | 261 (20)  | 268 (20)  |
|                     | 303 (20)  | 308 (20)  | 332 (20)  | 374*(20)  | 377 (20)  | 406 (20)  | 419 (20)  | 441 (20)  |
|                     | 450 (20)  | 465 (20)  | 466 (20)  | 502 (20)  | 509 (20)  | 592*(20)  | 595 (20)  | 595 (20)  |
|                     | 612 (20)  | 626 (20)  | 626 (20)  | 628 (20)  | 642*(20)  | 644 (20)  | 101 (21)  | 235*(21)  |
|                     | 235 (21)  | 235 (21)  | 264 (21)  | 385 (21)  | 443 (21)  | 541 (21)  | 977 (21)  | 103*(23)  |
|                     | 121 (23)  | 257 (23)  | 160*(25)  | 679 (26)  | 679*(26)  | 680 (26)  | 680*(26)  | 983 (26)  |
|                     | 1048 (26) | 1175 (26) | 1273 (26) | 266 (37)  | 269 (37)  | 283*(37)  | 456 (37)  | 484 (37)  |
|                     | 544*(37)  | 556 (37)  | 556 (37)  | 558 (37)  | 558 (37)  | 755 (37)  | 791 (37)  | 792 (37)  |
|                     | 798 (37)  | 803 (37)  | 806 (37)  | 810 (37)  | 814 (37)  | 818 (37)  | 887 (37)  | 941 (37)  |
|                     | 953 (37)  | 1110 (37) | 1256 (37) | 1853 (37) | 1888 (37) | 1899 (37) | 1913 (37) | 195 (39)  |
|                     | 201 (39)  | 215 (39)  | 217 (39)  | 285 (40)  | 382 (40)  | 383*(40)  | 383 (40)  | 534 (40)  |
|                     | 659*(40)  | 659 (40)  | 940 (40)  | 1222 (40) | 1270 (40) | 1279 (40) | 1279 (40) | 1321 (40) |
|                     | 39*(42)   |           |           |           |           |           |           |           |
| val                 | 455*(25)  | 473*(25)  | 1194*(26) | 1200 (26) | 1208*(26) | 1211 (26) |           |           |
| Validate            | 620*( 5)  | 662*( 5)  | 396 ( 6)  | 3364 ( 6) | 3379*( 6) | 3382*( 6) | 3509*( 6) | 3522*( 6) |
|                     | 3528*( 6) | 3532*( 6) | 2135*(13) | 492*(21)  | 494*(21)  |           |           |           |
| ValidatePrintRecord | 393*(36)  | 532 (37)  | 1731 (37) | 1823 (37) | 1955 (37) | 1975 (37) | 2058*(37) | 2076 (37) |
| validFInfo          | 1266*(13) | 616*(17)  | 685 (17)  |           |           |           |           |           |

|                |           |           |           |           |           |           |           |           |  |
|----------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|--|
| validInfo      | 1252*(17) | 1257=(17) | 1272 (17) | 1292*(17) | 1345=(17) | 1352 (17) |           |           |  |
| validPrefix    | 3516*( 6) | 3524 ( 6) | 3525 ( 6) |           |           |           |           |           |  |
| ValidRect      | 1774 (18) | 1330 (21) | 268 (23)  |           |           |           |           |           |  |
| ValidVRect     | 1606*(13) | 1765*(18) |           |           |           |           |           |           |  |
| value          | 633*( 5)  | 3425 ( 6) | 3454=( 6) | 2216*(13) | 2910*(13) | 714*(14)  | 720*(14)  | 755 (14)  |  |
|                | 755 (14)  | 771 (14)  | 774 (14)  | 973 (21)  | 1004=(21) |           |           |           |  |
| ValueAt        | 67*( 1)   | 318*( 2)  | 342=( 2)  | 347=( 2)  |           |           |           |           |  |
| valueStr       | 67*( 1)   | 69*( 1)   | 75*( 1)   | 79*( 1)   | 83*( 1)   | 162*( 2)  | 170 ( 2)  | 190*( 2)  |  |
|                | 220 ( 2)  | 228 ( 2)  | 236*( 2)  | 247 ( 2)  | 295*( 2)  | 306 ( 2)  | 318*( 2)  | 341=( 2)  |  |
|                | 346=( 2)  | 112*( 5)  | 633*( 6)  | 635 ( 6)  |           |           |           |           |  |
| vblAddr        | 137=( 4)  |           |           |           |           |           |           |           |  |
| vblCount       | 138=( 4)  | 218=( 4)  | 240=( 4)  |           |           |           |           |           |  |
| vblPhase       | 139=( 4)  |           |           |           |           |           |           |           |  |
| vblQElem       | 134 ( 4)  | 218 ( 4)  | 240 ( 4)  |           |           |           |           |           |  |
| vCentered      | 1889*(13) |           |           |           |           |           |           |           |  |
| vConstrain     | 1751*(13) | 64 (20)   | 99=(20)   |           |           |           |           |           |  |
| VCoordinate    | 604 ( 5)  | 606 ( 5)  | 3218 ( 6) | 3231 ( 6) | 600 ( 9)  | 2693 (10) | 1566 (13) | 1574 (13) |  |
|                | 1675 (13) | 1676 (13) | 1699 (13) | 1771 (13) | 1809 (13) | 1813 (13) | 1821 (13) | 1835 (13) |  |
|                | 1837 (13) | 1837 (13) | 1839 (13) | 1841 (13) | 1961 (13) | 1967 (13) | 2120 (13) | 2190 (13) |  |
|                | 2226 (13) | 2227 (13) | 2228 (13) | 2246 (13) | 2251 (13) | 2253 (13) | 2255 (13) | 2257 (13) |  |
|                | 2259 (13) | 2261 (13) | 2263 (13) | 2356 (13) | 2356 (13) | 2405 (13) | 2519 (13) | 621 (18)  |  |
|                | 707 (18)  | 708 (18)  | 767 (18)  | 1429 (18) | 1551 (18) | 761 (19)  | 816 (19)  | 15 (20)   |  |
|                | 374 (20)  | 397 (20)  | 434 (20)  | 435 (20)  | 457 (20)  | 528 (20)  | 528 (20)  | 532 (20)  |  |
|                | 551 (20)  | 556 (20)  | 592 (20)  | 605 (20)  | 637 (20)  | 424 (21)  | 777 (21)  | 1029 (21) |  |
|                | 1052 (21) | 1081 (21) | 1090 (21) | 1099 (21) | 1108 (21) | 1128 (21) | 1142 (21) | 1157 (21) |  |
|                | 34 (22)   | 24 (24)   | 25 (24)   | 101 (24)  | 156*(25)  | 160 (25)  | 161 (25)  | 166 (25)  |  |
|                | 894 (26)  | 219 (36)  | 220 (36)  | 232 (36)  | 240 (36)  | 428 (36)  | 407 (37)  | 408 (37)  |  |
|                | 413 (37)  | 445 (37)  | 725 (37)  | 770 (37)  | 778 (37)  | 840 (37)  | 842 (37)  | 843 (37)  |  |
|                | 844 (37)  | 971 (37)  | 243 (38)  | 335 (38)  | 336 (38)  | 573 (40)  | 574 (40)  | 1258 (40) |  |
|                | 39 (42)   | 50 (42)   | 52 (42)   | 54 (42)   | 64 (42)   |           |           |           |  |
| vDeltaOrg      | 1586*(13) | 579*(18)  | 589 (18)  |           |           |           |           |           |  |
| version        | 459 (16)  |           |           |           |           |           |           |           |  |
| versNum        | 459=(16)  |           |           |           |           |           |           |           |  |
| vertCenter     | 1879*(13) | 153 (19)  | 154 (19)  | 199=(19)  |           |           |           |           |  |
| vertConstrain  | 1822*(13) | 638*(20)  | 644 (20)  |           |           |           |           |           |  |
| vertically     | 2012*(13) | 346*(19)  | 354 (19)  | 364 (19)  |           |           |           |           |  |
| vertMax        | 1747*(13) | 61 (20)   | 95=(20)   |           |           |           |           |           |  |
| vertScrollUnit | 242*(40)  | 248=(40)  | 250=(40)  | 252=(40)  | 254 (40)  |           |           |           |  |
| vertUnits      | 1821*(13) | 637*(20)  | 642 (20)  |           |           |           |           |           |  |
| vh             | 852=(14)  | 852 (14)  | 859=(14)  | 864=(14)  | 864 (14)  | 634 (18)  | 636 (18)  | 638 (18)  |  |
|                | 641 (18)  | 644 (18)  | 653 (18)  | 653 (18)  | 661=(18)  | 1034 (18) | 1036=(18) | 1036 (18) |  |
|                | 1036 (18) | 1037=(18) | 1037 (18) | 1037 (18) | 1041=(18) | 1041 (18) | 1041 (18) | 1042=(18) |  |
|                | 1722=(18) | 1722 (18) | 1722 (18) | 166=(20)  | 166 (20)  | 166 (20)  | 167=(20)  | 167 (20)  |  |
|                | 167 (20)  | 168=(20)  | 168 (20)  | 168 (20)  | 169 (20)  | 170=(20)  | 195 (20)  | 195 (20)  |  |
|                | 196=(20)  | 196 (20)  | 196 (20)  | 197 (20)  | 197 (20)  | 198=(20)  | 198 (20)  | 199 (20)  |  |
|                | 199 (20)  | 262 (20)  | 263=(20)  | 263 (20)  | 263 (20)  | 264 (20)  | 265=(20)  | 265 (20)  |  |
|                | 266 (20)  | 266 (20)  | 443 (20)  | 444 (20)  | 444 (20)  | 446 (20)  | 446 (20)  | 447=(20)  |  |
|                | 535 (20)  | 536 (20)  | 536 (20)  | 537 (20)  | 542=(20)  | 542 (20)  | 559 (20)  | 564=(20)  |  |
|                | 568 (20)  | 571=(20)  | 583=(20)  | 583 (20)  | 585 (20)  | 614 (20)  | 614 (20)  | 615 (20)  |  |
|                | 617=(20)  | 621 (20)  | 622=(20)  | 124=(23)  | 124 (23)  | 124 (23)  | 259=(23)  | 259 (23)  |  |
|                | 259 (23)  | 260=(23)  | 161*(25)  | 471 (26)  | 471 (26)  | 417 (37)  | 417 (37)  | 423 (37)  |  |
|                | 462=(37)  | 463 (37)  | 463 (37)  | 463 (37)  | 463 (37)  | 469=(37)  | 485=(37)  | 486 (37)  |  |
|                | 486 (37)  | 487 (37)  | 488 (37)  | 728 (37)  | 734 (37)  | 734 (37)  | 783=(37)  | 784=(37)  |  |
|                | 786=(37)  | 786 (37)  | 786 (37)  | 850 (37)  | 851 (37)  | 942 (37)  | 943=(37)  | 943 (37)  |  |
|                | 946=(37)  | 947=(37)  | 947 (37)  | 947 (37)  | 948=(37)  | 948 (37)  | 948 (37)  | 949=(37)  |  |
|                | 949 (37)  | 949 (37)  | 980 (37)  | 980 (37)  | 981 (37)  | 982 (37)  | 983 (37)  | 986 (37)  |  |
|                | 988 (37)  | 989 (37)  | 994 (37)  | 1000=(37) | 1001=(37) | 1235=(37) | 1235 (37) | 1236=(37) |  |
|                | 1237 (37) | 1237 (37) | 1258=(37) | 1258 (37) | 1258 (37) | 1261=(37) | 1262 (37) | 1263 (37) |  |
|                | 1266=(37) | 1855 (37) | 1855 (37) | 1856 (37) | 1856 (37) | 1890=(37) | 1890 (37) | 1890 (37) |  |
|                | 1891=(37) | 1891 (37) | 1891 (37) | 1900=(37) | 1900 (37) | 1900 (37) | 1914=(37) | 1914 (37) |  |
|                | 1915 (37) | 2039 (37) | 2039 (37) | 2040 (37) | 203 (39)  | 203 (39)  | 204 (39)  | 204 (39)  |  |
|                | 205 (39)  | 207 (39)  | 208=(39)  | 208 (39)  | 210=(39)  | 210 (39)  | 287=(40)  | 287 (40)  |  |
|                | 588 (40)  | 594 (40)  | 605 (40)  | 629 (40)  | 630 (40)  | 631 (40)  | 941 (40)  | 942=(40)  |  |
|                | 942 (40)  | 942 (40)  |           |           |           |           |           |           |  |
| vWhite         | 270*(40)  | 277=(40)  | 283 (40)  |           |           |           |           |           |  |
| vhs            | 1675*(13) | 1698*(13) | 1835*(13) | 1837*(13) | 2355*(13) | 2404*(13) | 467*(18)  | 470 (18)  |  |
|                | 622*(18)  | 630=(18)  | 632 (18)  | 634 (18)  | 636 (18)  | 638 (18)  | 641 (18)  | 644 (18)  |  |
|                | 653 (18)  | 653 (18)  | 661 (18)  | 707*(18)  | 710 (18)  | 766*(18)  | 769 (18)  | 1008*(18) |  |
|                | 1033=(18) | 1034 (18) | 1036 (18) | 1036 (18) | 1036 (18) | 1037 (18) | 1037 (18) | 1037 (18) |  |
|                | 1041 (18) | 1041 (18) | 1041 (18) | 1042 (18) | 1712*(18) | 1717=(18) | 1718 (18) | 1720 (18) |  |
|                | 1722 (18) | 1722 (18) | 1722 (18) | 151*(20)  | 158=(20)  | 160 (20)  | 163 (20)  | 166 (20)  |  |
|                | 166 (20)  | 166 (20)  | 167 (20)  | 168 (20)  | 168 (20)  | 168 (20)  | 169 (20)  | 185*(20)  |  |
|                | 194=(20)  | 195 (20)  | 195 (20)  | 196 (20)  | 196 (20)  | 196 (20)  | 197 (20)  | 197 (20)  |  |
|                | 198 (20)  | 198 (20)  | 199 (20)  | 199 (20)  | 257*(20)  | 261=(20)  | 262 (20)  | 263 (20)  |  |
|                | 263 (20)  | 263 (20)  | 264 (20)  | 265 (20)  | 265 (20)  | 266 (20)  | 266 (20)  | 400*(20)  |  |
|                | 406=(20)  | 407 (20)  | 407 (20)  | 407 (20)  | 409 (20)  | 410 (20)  | 413 (20)  | 419=(20)  |  |
|                | 420 (20)  | 421 (20)  | 433*(20)  | 441=(20)  | 443 (20)  | 443 (20)  | 444 (20)  | 444 (20)  |  |
|                | 446 (20)  | 446 (20)  | 447 (20)  | 528*(20)  | 535 (20)  | 535 (20)  | 536 (20)  | 536 (20)  |  |
|                | 537 (20)  | 542 (20)  | 542 (20)  | 551*(20)  | 559 (20)  | 564 (20)  | 568 (20)  | 569 (20)  |  |
|                | 571 (20)  | 583 (20)  | 583 (20)  | 585 (20)  | 606*(20)  | 612=(20)  | 614 (20)  | 614 (20)  |  |
|                | 615 (20)  | 617 (20)  | 618 (20)  | 619 (20)  | 621 (20)  | 622 (20)  | 77*(23)   | 121=(23)  |  |
|                | 124 (23)  | 124 (23)  | 124 (23)  | 233*(23)  | 257=(23)  | 259 (23)  | 259 (23)  | 259 (23)  |  |
|                | 260 (23)  | 24*(24)   | 100*(24)  | 288*(25)  | 468*(26)  | 471 (26)  | 471 (26)  | 218*(36)  |  |
|                | 231*(36)  | 239*(36)  | 427*(36)  | 407*(37)  | 415 (37)  | 422 (37)  | 438*(37)  | 456=(37)  |  |
|                | 458 (37)  | 462 (37)  | 468 (37)  | 469 (37)  | 481*(37)  | 484=(37)  | 485 (37)  | 486 (37)  |  |
|                | 486 (37)  | 487 (37)  | 488 (37)  | 714*(37)  | 735 (37)  | 755=(37)  | 757 (37)  | 759 (37)  |  |
|                | 769*(37)  | 783 (37)  | 784 (37)  | 786 (37)  | 786 (37)  | 786 (37)  | 797 (37)  | 839*(37)  |  |
|                | 850 (37)  | 851 (37)  | 861 (37)  | 934*(37)  | 941=(37)  | 942 (37)  | 943 (37)  | 943 (37)  |  |
|                | 946 (37)  | 947 (37)  | 947 (37)  | 947 (37)  | 948 (37)  | 948 (37)  | 948 (37)  | 949 (37)  |  |
|                | 949 (37)  | 949 (37)  | 971*(37)  | 978 (37)  | 982 (37)  | 983 (37)  | 986 (37)  | 988 (37)  |  |
|                | 989 (37)  | 994 (37)  | 997 (37)  | 1000 (37) | 1001 (37) | 1253*(37) | 1256=(37) | 1257 (37) |  |
|                | 1258 (37) | 1258 (37) | 1258 (37) | 1261 (37) | 1262 (37) | 1263 (37) | 1265 (37) | 1266 (37) |  |
|                | 1847*(37) | 1853=(37) | 1855 (37) | 1855 (37) | 1855 (37) | 1856 (37) | 1856 (37) | 1856 (37) |  |
|                | 1883*(37) | 1888=(37) | 1890 (37) | 1890 (37) | 1890 (37) | 1891 (37) | 1891 (37) | 1891 (37) |  |
|                | 1899=(37) | 1900 (37) | 1900 (37) | 1900 (37) | 1911*(37) | 1913=(37) | 1914 (37) | 1914 (37) |  |
|                | 1915 (37) | 335*(38)  | 182*(39)  | 201=(39)  | 203 (39)  | 203 (39)  | 204 (39)  | 204 (39)  |  |
|                | 205 (39)  | 207 (39)  | 208 (39)  | 208 (39)  | 210 (39)  | 210 (39)  | 271*(40)  | 285=(40)  |  |
|                | 286 (40)  | 287 (40)  | 287 (40)  | 371*(40)  | 373 (40)  | 573*(40)  | 585 (40)  | 596 (40)  |  |
|                | 643*(40)  | 936*(40)  | 940=(40)  | 941 (40)  | 942 (40)  | 942 (40)  | 942 (40)  | 1295*(40) |  |

|                       |            |            |            |            |            |            |            |            |  |
|-----------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
|                       | 64* (42)   |            |            |            |            |            |            |            |  |
| VHSelect              | 1466 (13)  | 1675 (13)  | 1698 (13)  | 1761 (13)  | 1763 (13)  | 1829 (13)  | 1835 (13)  | 1837 (13)  |  |
|                       | 2225 (13)  | 2233 (13)  | 2286 (13)  | 2355 (13)  | 2404 (13)  | 2739 (13)  | 2739 (13)  | 2854 (13)  |  |
|                       | 832 (14)   | 467 (18)   | 622 (18)   | 707 (18)   | 766 (18)   | 1008 (18)  | 1712 (18)  | 151 (20)   |  |
|                       | 153 (20)   | 185 (20)   | 257 (20)   | 350 (20)   | 400 (20)   | 401 (20)   | 433 (20)   | 528 (20)   |  |
|                       | 551 (20)   | 606 (20)   | 943 (21)   | 1207 (21)  | 77 (23)    | 233 (23)   | 24 (24)    | 100 (24)   |  |
|                       | 161 (25)   | 288 (25)   | 468 (26)   | 103 (36)   | 150 (36)   | 218 (36)   | 231 (36)   | 239 (36)   |  |
|                       | 427 (36)   | 407 (37)   | 412 (37)   | 439 (37)   | 481 (37)   | 714 (37)   | 715 (37)   | 769 (37)   |  |
|                       | 839 (37)   | 934 (37)   | 971 (37)   | 976 (37)   | 1228 (37)  | 1253 (37)  | 1847 (37)  | 1883 (37)  |  |
|                       | 1911 (37)  | 2035 (37)  | 335 (38)   | 182 (39)   | 271 (40)   | 371 (40)   | 573 (40)   | 576 (40)   |  |
|                       | 643 (40)   | 936 (40)   | 1295 (40)  | 64 (42)    |            |            |            |            |  |
| view                  | 3102* (16) | 3126 (16)  | 3128 (16)  | 3179- (16) | 3181 (16)  | 3182 (16)  | 3206 (16)  | 3238 (16)  |  |
|                       | 3249 (16)  | 3252 (16)  | 3275 (16)  | 3276 (16)  | 3339 (16)  | 3342 (16)  | 976* (17)  | 978 (17)   |  |
|                       | 979 (17)   |            |            |            |            |            |            |            |  |
| viewArea              | 695* (10)  | 702 (10)   | 707 (10)   | 717* (37)  | 728 (37)   | 734 (37)   | 753 (37)   |            |  |
| viewarea              | 703 (10)   | 703 (10)   | 704 (10)   | 704 (10)   |            |            |            |            |  |
| viewedRect            | 1846* (37) | 1855 (37)  | 1856 (37)  | 1860 (37)  | 1861 (37)  | 1868 (37)  |            |            |  |
| ViewEnable            | 1147 ( 6)  | 1422 ( 6)  | 1645 ( 6)  | 1859 ( 6)  | 2697 ( 6)  | 2940 ( 6)  | 1653* (13) | 1782* (18) |  |
| viewExtent            | 3116* (16) | 3182 (16)  | 3207 (16)  |            |            |            |            |            |  |
| viewPerPage           | 1687* (13) | 727* (18)  | 729- (18)  | 730 (18)   | 1623* (37) | 1641 (37)  | 1642 (37)  | 1657 (37)  |  |
|                       | 1662 (37)  | 340* (38)  | 641* (40)  | 646 (40)   | 652 (40)   | 659 (40)   | 659 (40)   | 665 (40)   |  |
| viewPt                | 1494* (13) | 1498* (13) | 1843* (13) | 1516* (18) | 1518 (18)  | 1519 (18)  | 1793* (18) | 1795 (18)  |  |
|                       | 1796 (18)  | 182* (20)  | 195 (20)   | 197 (20)   | 179* (39)  | 197 (39)   | 198 (39)   |            |  |
| viewRect              | 2624 ( 6)  | 1496* (13) | 1500* (13) | 1604* (13) | 1606* (13) | 1005* (18) | 1026 (18)  | 1029 (18)  |  |
|                       | 1036 (18)  | 1355* (18) | 1362 (18)  | 1526* (18) | 1528 (18)  | 1529 (18)  | 1765* (18) | 1772 (18)  |  |
|                       | 1803* (18) | 1805 (18)  | 1806 (18)  | 1565- (40) |            |            |            |            |  |
| ViewResource          | 1453 (13)  | 1454* (13) |            |            |            |            |            |            |  |
| viewRsrc              | 2931* (13) | 2935* (13) | 2942* (13) | 64* (18)   | 67 (18)    | 68 (18)    | 68 (18)    | 97* (18)   |  |
|                       | 107 (18)   | 108 (18)   | 121 (18)   | 122 (18)   | 124 (18)   | 125 (18)   | 126 (18)   | 136* (18)  |  |
|                       | 139 (18)   |            |            |            |            |            |            |            |  |
| ViewRsrcHndl          | 77 ( 5)    | 79 ( 5)    | 151 ( 5)   | 153 ( 5)   | 181 ( 5)   | 183 ( 5)   | 221 ( 5)   | 223 ( 5)   |  |
|                       | 264 ( 5)   | 266 ( 5)   | 316 ( 5)   | 318 ( 5)   | 362 ( 5)   | 364 ( 5)   | 407 ( 5)   | 409 ( 5)   |  |
|                       | 456 ( 5)   | 458 ( 5)   | 523 ( 5)   | 525 ( 5)   | 577 ( 5)   | 579 ( 5)   | 653 ( 5)   | 655 ( 5)   |  |
|                       | 345 ( 6)   | 367 ( 6)   | 819 ( 6)   | 844 ( 6)   | 901 ( 6)   | 924 ( 6)   | 1028 ( 6)  | 1051 ( 6)  |  |
|                       | 1178 ( 6)  | 1200 ( 6)  | 1481 ( 6)  | 1509 ( 6)  | 1703 ( 6)  | 1728 ( 6)  | 1907 ( 6)  | 1928 ( 6)  |  |
|                       | 2139 ( 6)  | 2165 ( 6)  | 2731 ( 6)  | 2757 ( 6)  | 2974 ( 6)  | 2995 ( 6)  | 3441 ( 6)  | 3464 ( 6)  |  |
|                       | 199 ( 9)   | 202 ( 9)   | 457 ( 9)   | 460 ( 9)   | 542 ( 9)   | 240 (10)   | 277 (10)   | 2270 (10)  |  |
|                       | 2302 (10)  | 2472 (10)  | 1452* (13) | 1484 (13)  | 1486 (13)  | 1780 (13)  | 1782 (13)  | 1940 (13)  |  |
|                       | 1942 (13)  | 2082 (13)  | 2084 (13)  | 2161 (13)  | 2163 (13)  | 2241 (13)  | 2243 (13)  | 2295 (13)  |  |
|                       | 2297 (13)  | 2927 (13)  | 110 (15)   | 129 (15)   | 41 (18)    | 44 (18)    | 48 (18)    | 239 (18)   |  |
|                       | 275 (18)   | 171 (19)   | 214 (19)   | 80 (20)    | 111 (20)   | 162 (21)   | 204 (21)   | 567 (21)   |  |
|                       | 579 (21)   | 991 (21)   | 1017 (21)  | 1273 (21)  | 1285 (21)  | 165 (38)   | 167 (38)   | 127 (40)   |  |
|                       | 164 (40)   |            |            |            |            |            |            |            |  |
| viewRsrcID            | 2871* (13) | 1020* (14) | 1027 (14)  |            |            |            |            |            |  |
| ViewRsrcPtr           | 1452 (13)  | 1453* (13) |            |            |            |            |            |            |  |
| ViewTemplate          | 1435 (13)  | 1436* (13) | 1456 (13)  | 177 (15)   | 230 (18)   | 245 (18)   |            |            |  |
| ViewTemplatePtr       | 1435* (13) | 68 (15)    | 90 (15)    | 111 (15)   | 222 (18)   | 241 (18)   | 244 (18)   |            |  |
| viewThatHandledCursor | 1267* (18) | 1277- (18) | 1278 (18)  | 1283- (18) | 1286 (18)  | 1288- (18) | 1290 (18)  |            |  |
| viewThatHandledMouse  | 1302* (18) | 1327- (18) | 1329 (18)  |            |            |            |            |            |  |
| viewToDelete          | 1415* (13) | 222* (17)  | 224 (17)   |            |            |            |            |            |  |
| ViewToQDpt            | 1498* (13) | 1182 (18)  | 1183 (18)  | 1287 (18)  | 1333 (18)  | 1793* (18) | 1796- (18) | 87 (22)    |  |
|                       | 87 (22)    | 785 (37)   | 787 (37)   |            |            |            |            |            |  |
| ViewToQDRect          | 500 (10)   | 525 (10)   | 717 (10)   | 2950 (10)  | 1500* (13) | 1032 (18)  | 1217 (18)  | 1362 (18)  |  |
|                       | 1580 (18)  | 1585 (18)  | 1772 (18)  | 1803* (18) | 490 (20)   | 960 (37)   |            |            |  |
| VInstall              | 149 ( 4)   |            |            |            |            |            |            |            |  |
| visible               | 1218* (16) | 1268- (16) | 751 (19)   |            |            |            |            |            |  |
| VisibleRect           | 155* (18)  | 1218 (18)  | 1346 (18)  | 1363 (18)  | 1581 (18)  | 1586 (18)  | 1773 (18)  |            |  |
| visQDRect             | 1651* (13) | 1209* (18) | 1217 (18)  | 1218 (18)  |            |            |            |            |  |
| visRect               | 874* (18)  | 906 (18)   | 908 (18)   | 908 (18)   | 910 (18)   | 916 (18)   | 1006* (18) | 1029 (18)  |  |
|                       | 1036 (18)  | 180* (39)  | 199 (39)   | 203 (39)   | 205 (39)   | 1300* (40) | 1306 (40)  | 1312 (40)  |  |
|                       | 1317 (40)  |            |            |            |            |            |            |            |  |
| visRgn                | 1145 (14)  | 20 (18)    | 148 (18)   | 898 (18)   | 908 (18)   | 83 (39)    | 88 (39)    |            |  |
| vNewSelRect           | 1301* (40) | 1330 (40)  | 1331 (40)  |            |            |            |            |            |  |
| volRefNum             | 1287* (13) | 904* (17)  |            |            |            |            |            |            |  |
| volRefnum             | 829* (13)  | 1192* (13) | 1312* (13) | 1318* (13) | 1334* (13) | 270* (16)  | 307 (16)   | 720* (17)  |  |
|                       | 725 (17)   | 933- (17)  | 938 (17)   | 944 (17)   | 1041* (17) | 1066 (17)  | 1087 (17)  | 1091- (17) |  |
|                       | 1103 (17)  | 1114 (17)  | 1133 (17)  | 1144 (17)  | 1173 (17)  | 1181 (17)  | 1196 (17)  | 1198 (17)  |  |
|                       | 1219* (17) | 1230 (17)  | 1232 (17)  | 1234 (17)  | 1249* (17) | 1263 (17)  | 1270 (17)  | 1289* (17) |  |
|                       | 1303 (17)  | 1350 (17)  | 1365 (17)  | 1370 (17)  | 386* (25)  | 390* (25)  | 406* (25)  | 413* (25)  |  |
|                       | 133* (26)  | 140 (26)   | 172* (26)  | 176 (26)   | 233* (26)  | 239 (26)   | 579* (26)  | 605 (26)   |  |
|                       | 638 (26)   | 653 (26)   |            |            |            |            |            |            |  |
| VPoint                | 67 ( 5)    | 141 ( 5)   | 171 ( 5)   | 211 ( 5)   | 254 ( 5)   | 306 ( 5)   | 352 ( 5)   | 397 ( 5)   |  |
|                       | 446 ( 5)   | 513 ( 5)   | 568 ( 5)   | 644 ( 5)   | 254 ( 5)   | 780 ( 6)   | 861 ( 6)   | 989 ( 6)   |  |
|                       | 1119 ( 6)  | 1390 ( 6)  | 1613 ( 6)  | 1833 ( 6)  | 2024 ( 6)  | 2668 ( 6)  | 2925 ( 6)  | 3395 ( 6)  |  |
|                       | 174 ( 9)   | 217 ( 9)   | 388 ( 9)   | 430 ( 9)   | 525 ( 9)   | 653 ( 9)   | 686 ( 9)   | 691 ( 9)   |  |
|                       | 32 (10)    | 44 (10)    | 351 (10)   | 1452 (10)  | 2072 (10)  | 2202 (10)  | 2444 (10)  | 2820 (10)  |  |
|                       | 2930 (10)  | 3029 (10)  | 1439 (13)  | 1440 (13)  | 1464 (13)  | 1465 (13)  | 1477 (13)  | 1478 (13)  |  |
|                       | 1494 (13)  | 1498 (13)  | 1502 (13)  | 1506 (13)  | 1510 (13)  | 1514 (13)  | 1562 (13)  | 1564 (13)  |  |
|                       | 1568 (13)  | 1570 (13)  | 1619 (13)  | 1621 (13)  | 1630 (13)  | 1687 (13)  | 1713 (13)  | 1758 (13)  |  |
|                       | 1759 (13)  | 1760 (13)  | 1769 (13)  | 1799 (13)  | 1801 (13)  | 1817 (13)  | 1819 (13)  | 1831 (13)  |  |
|                       | 1833 (13)  | 1843 (13)  | 1843 (13)  | 1987 (13)  | 2073 (13)  | 2089 (13)  | 2091 (13)  | 2122 (13)  |  |
|                       | 2123 (13)  | 2128 (13)  | 2132 (13)  | 2153 (13)  | 2231 (13)  | 2284 (13)  | 2334 (13)  | 2367 (13)  |  |
|                       | 2418 (13)  | 2480 (13)  | 2481 (13)  | 2486 (13)  | 2493 (13)  | 2535 (13)  | 2538 (13)  | 2541 (13)  |  |
|                       | 2542 (13)  | 2571 (13)  | 2573 (13)  | 2702 (13)  | 2740 (13)  | 2765 (13)  | 2815 (13)  | 836 (14)   |  |
|                       | 837 (14)   | 903 (14)   | 1557 (16)  | 3006 (16)  | 3106 (16)  | 3107 (16)  | 3108 (16)  | 3111 (16)  |  |
|                       | 3113 (16)  | 3115 (16)  | 179 (18)   | 179 (18)   | 378 (18)   | 555 (18)   | 570 (18)   | 619 (18)   |  |
|                       | 623 (18)   | 681 (18)   | 727 (18)   | 832 (18)   | 873 (18)   | 877 (18)   | 1002 (18)  | 1263 (18)  |  |
|                       | 1271 (18)  | 1297 (18)  | 1309 (18)  | 1402 (18)  | 1411 (18)  | 1516 (18)  | 1554 (18)  | 1683 (18)  |  |
|                       | 1699 (18)  | 1708 (18)  | 1710 (18)  | 1793 (18)  | 1813 (18)  | 684 (19)   | 13 (20)    | 18 (20)    |  |
|                       | 49 (20)    | 154 (20)   | 155 (20)   | 182 (20)   | 182 (20)   | 210 (20)   | 254 (20)   | 364 (20)   |  |
|                       | 436 (20)   | 460 (20)   | 479 (20)   | 531 (20)   | 555 (20)   | 602 (20)   | 607 (20)   | 651 (20)   |  |
|                       | 661 (20)   | 54 (21)    | 55 (21)    | 65 (21)    | 76 (21)    | 97 (21)    | 226 (21)   | 248 (21)   |  |
|                       | 451 (21)   | 452 (21)   | 460 (21)   | 470 (21)   | 532 (21)   | 941 (21)   | 1205 (21)  | 71 (22)    |  |
|                       | 72 (22)    | 81 (22)    | 98 (22)    | 17 (23)    | 18 (23)    | 74 (23)    | 297 (23)   | 68 (24)    |  |
|                       | 220 (24)   | 158* (25)  | 167 (25)   | 464 (25)   | 482 (25)   | 895 (26)   | 1270 (26)  | 1280 (26)  |  |
|                       | 186 (36)   | 214 (36)   | 255 (36)   | 353 (36)   | 479 (37)   | 773 (37)   | 1247 (37)  | 1623 (37)  |  |
|                       | 1909 (37)  | 146 (38)   | 147 (38)   | 233 (38)   | 240 (38)   | 340 (38)   | 343 (38)   | 179 (39)   |  |
|                       | 181 (39)   | 14 (40)    | 265 (40)   | 366 (40)   | 641 (40)   | 934 (40)   | 1262 (40)  | 351 (41)   |  |
|                       | 31 (42)    | 33 (42)    | 35 (42)    | 35 (42)    | 37 (42)    | 37 (42)    | 39 (42)    | 41 (42)    |  |

|                   |            |            |            |            |            |            |            |            |  |
|-------------------|------------|------------|------------|------------|------------|------------|------------|------------|--|
| VPointToCell      | 56 (42)    | 66 (42)    | 70 (42)    |            |            |            |            |            |  |
| vPt               | 388* (9)   | 709 (10)   | 710 (10)   | 1457 (10)  | 2072* (10) | 2129* (10) | 2941 (10)  | 2992 (10)  |  |
| vPtToPt           | 773* (37)  | 783 (37)   | 784 (37)   | 785 (37)   | 786 (37)   | 787 (37)   | 39* (42)   |            |  |
| VRect             | 1796 (18)  | 253 (21)   | 33* (42)   |            |            |            |            |            |  |
|                   | 3316 (6)   | 3333 (6)   | 265 (9)    | 269 (9)    | 270 (9)    | 395 (10)   | 397 (10)   | 432 (10)   |  |
|                   | 490 (10)   | 491 (10)   | 546 (10)   | 547 (10)   | 565 (10)   | 566 (10)   | 689 (10)   | 690 (10)   |  |
|                   | 695 (10)   | 845 (10)   | 942 (10)   | 1406 (10)  | 1422 (10)  | 1423 (10)  | 1451 (10)  | 1521 (10)  |  |
|                   | 1621 (10)  | 1786 (10)  | 1857 (10)  | 2876 (10)  | 2877 (10)  | 2878 (10)  | 2879 (10)  | 2935 (10)  |  |
|                   | 1496 (13)  | 1500 (13)  | 1604 (13)  | 1606 (13)  | 1610 (13)  | 1645 (13)  | 1649 (13)  | 1720 (13)  |  |
|                   | 1764 (13)  | 1805 (13)  | 1845 (13)  | 2816 (13)  | 905 (14)   | 3116 (16)  | 3117 (16)  | 684 (18)   |  |
|                   | 885 (18)   | 1005 (18)  | 1006 (18)  | 1007 (18)  | 1078 (18)  | 1136 (18)  | 1146 (18)  | 1171 (18)  |  |
|                   | 1212 (18)  | 1355 (18)  | 1526 (18)  | 1556 (18)  | 1613 (18)  | 1625 (18)  | 1765 (18)  | 1803 (18)  |  |
|                   | 186 (20)   | 328 (20)   | 428 (20)   | 432 (20)   | 482 (20)   | 164* (25)  | 467 (25)   | 485 (25)   |  |
|                   | 896 (26)   | 1292 (26)  | 1304 (26)  | 97 (36)    | 188 (36)   | 717 (37)   | 1774 (37)  | 1846 (37)  |  |
|                   | 347 (38)   | 1059 (40)  | 1301 (40)  | 46 (42)    | 48 (42)    | 50 (42)    | 52 (42)    | 54 (42)    |  |
|                   | 56 (42)    | 58 (42)    | 58 (42)    | 60 (42)    | 62 (42)    | 64 (42)    | 66 (42)    | 68 (42)    |  |
|                   | 68 (42)    | 70 (42)    |            |            |            |            |            |            |  |
| VRectToRect       | 897 (18)   | 1806 (18)  | 101 (20)   | 48* (42)   |            |            |            |            |  |
| vRefnum           | 361 (16)   | 457* (16)  | 457 (16)   | 2215 (16)  | 827* (17)  | 832 (17)   | 848 (17)   | 893 (17)   |  |
|                   | 933 (17)   | 402* (25)  | 204* (26)  | 213 (26)   | 216 (26)   | 219* (26)  |            |            |  |
| VRemove           | 183 (4)    |            |            |            |            |            |            |            |  |
| vScrollUnits      | 1749* (13) | 64 (20)    | 97* (20)   |            |            |            |            |            |  |
| vStrip            | 252* (36)  | 2031* (37) | 2037 (37)  |            |            |            |            |            |  |
| vType             | 136 (4)    |            |            |            |            |            |            |            |  |
| vwPtr             | 241* (18)  | 244* (18)  | 247 (18)   |            |            |            |            |            |  |
| -W-               |            |            |            |            |            |            |            |            |  |
| w                 | 2847* (13) | 273* (14)  | 277 (14)   | 279 (14)   |            |            |            |            |  |
| WaitNextEvent     | 1160 (16)  |            |            |            |            |            |            |            |  |
| wantHorzSBar      | 1772* (13) | 1826* (13) | 16* (20)   | 36 (20)    | 206* (20)  | 223 (20)   |            |            |  |
| wantHSBar         | 1746* (13) | 68 (20)    | 94* (20)   |            |            |            |            |            |  |
| wantHScrollBar    | 2852* (13) | 2860* (13) | 830* (14)  | 860 (14)   | 868 (14)   | 898* (14)  | 922 (14)   | 926 (14)   |  |
|                   | 935 (14)   | 941 (14)   |            |            |            |            |            |            |  |
| wantsTheMouse     | 690* (19)  | 696* (19)  | 705* (19)  | 708 (19)   |            |            |            |            |  |
| wantValidate      | 1793* (37) | 1800* (37) | 1814* (37) | 1822 (37)  |            |            |            |            |  |
| wantVertSBar      | 1772* (13) | 1826* (13) | 16* (20)   | 36 (20)    | 206* (20)  | 235 (20)   |            |            |  |
| wantVBar          | 1745* (13) | 68 (20)    | 93* (20)   |            |            |            |            |            |  |
| wantVScrollBar    | 2852* (13) | 2860* (13) | 830* (14)  | 860 (14)   | 868 (14)   | 898* (14)  | 922 (14)   | 929 (14)   |  |
|                   | 932 (14)   | 941 (14)   |            |            |            |            |            |            |  |
| wasActive         | 602* (5)   | 681* (5)   | 2616* (6)  | 2628 (6)   | 3312* (6)  | 3322 (6)   | 658* (13)  | 313* (15)  |  |
|                   | 205* (38)  | 1178* (40) |            |            |            |            |            |            |  |
| wasEnabled        | 425* (30)  | 442* (30)  | 484 (30)   |            |            |            |            |            |  |
| wasVisible        | 893* (21)  | 904* (21)  | 907 (21)   |            |            |            |            |            |  |
| watchCursor       | 28* (4)    | 116* (4)   | 116 (4)    | 128 (4)    | 322 (4)    |            |            |            |  |
| watchDelay        | 15* (4)    | 73 (4)     | 94* (4)    | 122* (4)   | 240 (4)    | 274 (4)    |            |            |  |
| watchOn           | 26* (4)    | 121* (4)   | 213 (4)    | 239* (4)   | 328* (4)   |            |            |            |  |
| wDev              | 1725 (37)  |            |            |            |            |            |            |            |  |
| WDPBRec           | 205 (26)   |            |            |            |            |            |            |            |  |
| what              | 635 (16)   | 1167 (16)  | 1188 (16)  | 1192* (16) | 1404 (16)  | 1409 (16)  | 2557 (16)  | 2564 (16)  |  |
|                   | 2569 (16)  | 2577 (16)  | 2579 (16)  | 2596 (16)  | 2631 (16)  | 91 (37)    | 91 (37)    |            |  |
| WhatToDo          | 265* (36)  | 624* (37)  | 642 (37)   |            |            |            |            |            |  |
| when              | 591 (16)   | 1648 (16)  | 2542 (16)  |            |            |            |            |            |  |
| where             | 787* (13)  | 870* (13)  | 1348* (13) | 2834* (13) | 1051* (14) | 1054 (14)  | 333* (16)  | 349 (16)   |  |
|                   | 415* (16)  | 427 (16)   | 450 (16)   | 592 (16)   | 595 (16)   | 1565 (16)  | 1587 (16)  | 1601 (16)  |  |
|                   | 1610 (16)  | 1617 (16)  | 1621 (16)  | 1624 (16)  | 1627 (16)  | 1631 (16)  | 2573 (16)  | 2573 (16)  |  |
|                   | 2645* (16) | 2649 (16)  | 2668 (16)  | 2927* (16) | 2934* (16) | 3338 (16)  | 1400* (17) | 1406* (17) |  |
| whereMouseDown    | 977* (13)  | 578* (16)  | 589 (16)   | 598 (16)   | 1553* (16) | 1565* (16) | 1566 (16)  | 1570 (16)  |  |
|                   | 1575 (16)  | 1576 (16)  | 1587 (16)  | 1593 (16)  | 1597 (16)  | 1631 (16)  | 1632 (16)  |            |  |
| which             | 44* (38)   |            |            |            |            |            |            |            |  |
| whichBreak        | 1698* (13) | 2404* (13) | 766* (18)  | 769 (18)   | 100* (24)  | 240* (36)  | 427* (36)  | 716* (37)  |  |
|                   | 733* (37)  | 733 (37)   | 735 (37)   | 758* (37)  | 769* (37)  | 805 (37)   | 809 (37)   | 813 (37)   |  |
|                   | 817 (37)   | 971* (37)  | 980 (37)   | 982 (37)   | 986 (37)   | 996 (37)   | 1000 (37)  |            |  |
| whichItem         | 34* (30)   | 406* (30)  |            |            |            |            |            |            |  |
| whichPart         | 623* (10)  | 633* (10)  | 637 (10)   |            |            |            |            |            |  |
| whichWay          | 2854* (13) | 832* (14)  | 852 (14)   | 852 (14)   | 859 (14)   | 864 (14)   | 864 (14)   |            |  |
| WhileFocused      | 2202* (13) | 669 (21)   | 769 (21)   | 790 (21)   | 809 (21)   | 827 (21)   | 845 (21)   | 863 (21)   |  |
|                   | 883 (21)   | 890* (21)  | 1334 (21)  |            |            |            |            |            |  |
| white             | 1213 (14)  |            |            |            |            |            |            |            |  |
| whiteColor        | 135 (6)    | 806 (26)   |            |            |            |            |            |            |  |
| who               | 150* (8)   | 160* (8)   | 162* (8)   | 164 (8)    |            |            |            |            |  |
| whoCares          | 709* (41)  | 777* (41)  |            |            |            |            |            |            |  |
| wholeRect         | 275* (9)   | 488* (10)  | 510 (10)   |            |            |            |            |            |  |
| whoPC             | 159* (7)   | 158 (8)    |            |            |            |            |            |            |  |
| wid               | 2367* (6)  | 2381* (6)  | 2383 (6)   | 2386* (6)  | 2386 (6)   | 2391 (6)   |            |            |  |
| widMax            | 1561 (40)  |            |            |            |            |            |            |            |  |
| width             | 606* (5)   | 3231* (6)  | 3234 (6)   | 3236 (6)   | 600* (9)   | 434* (10)  | 453* (10)  | 459* (10)  |  |
|                   | 467* (10)  | 467 (10)   | 476 (10)   | 2693* (10) | 2695 (10)  | 2698 (10)  | 2699 (10)  | 1566* (13) |  |
|                   | 1809* (13) | 1967* (13) | 2120* (13) | 2190* (13) | 1551* (18) | 1568 (18)  | 1570 (18)  | 1572 (18)  |  |
|                   | 816* (19)  | 822 (19)   | 824 (19)   | 835 (19)   | 839 (19)   | 397* (20)  | 415 (20)   | 424* (21)  |  |
|                   | 429 (21)   | 777* (21)  | 783 (21)   | 791 (21)   | 243* (38)  | 1258* (40) | 1265 (40)  |            |  |
| window            | 735* (17)  | 744* (17)  | 745 (17)   |            |            |            |            |            |  |
| windowBounds      | 3007* (16) | 3043 (16)  | 3044 (16)  | 3045 (16)  | 3065 (16)  | 3066 (16)  |            |            |  |
| WindowFlags       | 1886* (13) |            |            |            |            |            |            |            |  |
| windowKind        | 545 (16)   | 1874 (16)  |            |            |            |            |            |            |  |
| WindowPeek        | 545 (16)   | 865 (16)   | 1874 (16)  | 2601 (16)  | 751 (19)   | 880 (19)   | 273 (20)   | 646 (21)   |  |
| WindowPtr         | 758 (13)   | 916 (13)   | 922 (13)   | 927 (13)   | 962 (13)   | 1905 (13)  | 1931 (13)  | 2766 (13)  |  |
|                   | 2847 (13)  | 273 (14)   | 976 (14)   | 501 (16)   | 540 (16)   | 841 (16)   | 1211 (16)  | 1227 (16)  |  |
|                   | 1276 (16)  | 1311 (16)  | 1316 (16)  | 1552 (16)  | 1662 (16)  | 1712 (16)  | 1717 (16)  | 1871 (16)  |  |
|                   | 2734 (16)  | 3003 (16)  | 3375 (16)  | 15 (19)    | 87 (19)    | 120 (19)   | 230 (19)   | 610 (21)   |  |
|                   | 615 (21)   | 693 (21)   | 700 (21)   |            |            |            |            |            |  |
| WindowRecord      | 2666 (13)  |            |            |            |            |            |            |            |  |
| WindowTemplate    | 1865 (13)  | 1866* (13) | 1215* (16) | 1223 (16)  | 160 (19)   | 183 (19)   |            |            |  |
| WindowTemplatePtr | 1865* (13) | 113 (19)   | 175 (19)   | 182 (19)   |            |            |            |            |  |
| windowToDelete    | 902* (13)  | 1418* (13) | 618* (16)  | 620 (16)   | 232* (17)  | 234 (17)   |            |            |  |
| WindowToLocal     | 1514* (13) | 1813* (18) |            |            |            |            |            |            |  |
| windowVPt         | 3006* (16) | 3044 (16)  | 3045 (16)  | 3046 (16)  |            |            |            |            |  |
| windRect          | 307* (19)  | 319 (19)   | 320 (19)   | 321 (19)   |            |            |            |            |  |
| wmgrPort          | 3124* (16) | 3150 (16)  | 3151 (16)  |            |            |            |            |            |  |
| WMgrToWindow      | 927* (13)  | 548 (16)   | 1316 (16)  | 1573 (16)  | 1688 (16)  | 1717 (16)  | 3031 (16)  | 3375* (16) |  |

|                  |           |           |           |           |           |           |           |           |
|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| wmgrWindow       | 3383-(16) | 3385-(16) |           |           |           |           |           |           |
| wName            | 501*(16)  | 523-(16)  | 524 (16)  | 525 (16)  | 526 (16)  | 230*(19)  | 234-(19)  | 245 (19)  |
| wPtr             | 2536*(16) | 2601-(16) | 2602 (16) |           |           |           |           |           |
| worryAboutBreaks | 175*(19)  | 182-(19)  | 185 (19)  |           |           |           |           |           |
| wouldPrintRect   | 1621*(37) | 1628-(37) | 1630 (37) | 1668 (37) |           |           |           |           |
| WRes             | 1139 (14) |           |           |           |           |           |           |           |
|                  | 77*( 5)   | 151*( 5)  | 181*( 5)  | 221*( 5)  | 264*( 5)  | 316*( 5)  | 362*( 5)  | 407*( 5)  |
|                  | 456*( 5)  | 523*( 5)  | 577*( 5)  | 653*( 5)  | 345*( 6)  | 351 ( 6)  | 370 ( 6)  | 819*( 6)  |
|                  | 826 ( 6)  | 847 ( 6)  | 901*( 6)  | 908 ( 6)  | 927 ( 6)  | 1028*( 6) | 1035 ( 6) | 1054 ( 6) |
|                  | 1178*( 6) | 1185 ( 6) | 1203 ( 6) | 1481*( 6) | 1487 ( 6) | 1512 ( 6) | 1703*( 6) | 1709 ( 6) |
|                  | 1731 ( 6) | 1907*( 6) | 1913 ( 6) | 1931 ( 6) | 2139*( 6) | 2145 ( 6) | 2168 ( 6) | 2731*( 6) |
|                  | 2738 ( 6) | 2760 ( 6) | 2974*( 6) | 2980 ( 6) | 2998 ( 6) | 3441*( 6) | 3447 ( 6) | 3467 ( 6) |
|                  | 199*( 9)  | 457*( 9)  | 240*(10)  | 246 (10)  | 280 (10)  | 2270*(10) | 2278 (10) | 2305 (10) |
|                  | 2475 (10) | 1484*(13) | 1780*(13) | 1940*(13) | 2082*(13) | 2161*(13) | 2241*(13) | 2295*(13) |
|                  | 239*(18)  | 278 (18)  | 171*(19)  | 178 (19)  | 217 (19)  | 80*(20)   | 86 (20)   | 114 (20)  |
|                  | 162*(21)  | 169 (21)  | 207 (21)  | 567*(21)  | 569 (21)  | 991*(21)  | 997 (21)  | 1020 (21) |
|                  | 1273*(21) | 1275 (21) | 165*(38)  | 127*(40)  | 132 (40)  | 167 (40)  |           |           |
| WRITE            | 233 ( 6)  | 233 ( 6)  | 236 ( 6)  | 236 ( 6)  | 1142 (16) | 1144 (16) | 171 (32)  | 174 (32)  |
| Write            | 129 (39)  | 146 (39)  | 651 (40)  | 664 (40)  | 1311 (40) | 1312 (40) | 1313 (40) |           |
|                  | 302 ( 8)  | 304 ( 8)  | 700 (14)  | 701 (14)  | 702 (14)  | 703 (14)  | 154 (15)  | 2542 (16) |
|                  | 2562 (16) | 2570 (16) | 2572 (16) | 2578 (16) | 2580 (16) | 2582 (16) | 2588 (16) | 2590 (16) |
|                  | 2597 (16) | 2599 (16) | 2608 (16) | 116 (23)  | 697 (26)  | 716 (26)  | 1175 (26) | 1185 (26) |
|                  | 1201 (26) | 1211 (26) | 1224 (26) | 1235 (26) | 1248 (26) | 1250 (26) | 1260 (26) | 1273 (26) |
|                  | 1283 (26) | 1296 (26) | 1307 (26) | 1323 (26) | 1332 (26) | 1346 (26) | 1355 (26) | 1369 (26) |
|                  | 1378 (26) | 336 (28)  | 342 (28)  | 350 (28)  | 357 (28)  | 364 (28)  | 1866 (37) | 1867 (37) |
|                  | 1869 (37) | 45 (39)   | 61 (39)   | 65 (39)   | 66 (39)   | 67 (39)   | 68 (39)   | 74 (39)   |
|                  | 83 (39)   | 84 (39)   | 88 (39)   | 89 (39)   |           |           |           |           |
| WriteBoolean     | 461*(25)  | 1244*(26) | 1261 (26) |           |           |           |           |           |
| WriteChar        | 36*(39)   | 77 (39)   |           |           |           |           |           |           |
| WriteFocus       | 14*(18)   |           |           |           |           |           |           |           |
| WriteHexInt      | 492*(25)  | 1341*(26) | 1355 (26) |           |           |           |           |           |
| WriteHexLongint  | 496*(25)  | 1364*(26) | 1378 (26) |           |           |           |           |           |
| Writeln          | 234 ( 6)  | 237 ( 6)  | 306 ( 8)  | 828 (10)  | 1100 (10) | 1101 (10) | 1102 (10) | 1103 (10) |
|                  | 2394 (10) | 2616 (10) | 148 (15)  | 202 (16)  | 1147 (16) | 3022 (16) | 3079 (16) | 3084 (16) |
|                  | 17 (18)   | 19 (18)   | 21 (18)   | 23 (18)   | 976 (18)  | 1866 (18) | 496 (20)  | 498 (20)  |
|                  | 576 (20)  | 166 (32)  | 200 (32)  | 110 (39)  | 114 (39)  | 115 (39)  | 119 (39)  | 121 (39)  |
|                  | 125 (39)  | 131 (39)  | 136 (39)  | 138 (39)  | 142 (39)  | 148 (39)  | 153 (39)  | 154 (39)  |
|                  | 338 (40)  | 653 (40)  | 666 (40)  | 1311 (40) | 1313 (40) | 1327 (40) | 1622 (40) | 865 (41)  |
|                  | 866 (41)  | 867 (41)  | 868 (41)  |           |           |           |           |           |
| WriteLn          | 1496 ( 6) | 1718 ( 6) | 1919 ( 6) | 251 ( 8)  | 556 (14)  | 558 (14)  | 560 (14)  | 562 (14)  |
|                  | 564 (14)  | 1054 (14) | 1126 (14) | 1274 (14) | 1275 (14) | 151 (15)  | 155 (15)  | 157 (15)  |
|                  | 158 (15)  | 252 (15)  | 1173 (16) | 1474 (16) | 1989 (16) | 2350 (16) | 2566 (16) | 2573 (16) |
|                  | 2592 (16) | 2602 (16) | 2605 (16) | 2613 (16) | 2615 (16) | 2617 (16) | 2619 (16) | 2627 (16) |
|                  | 2629 (16) | 2672 (16) | 1253 (18) | 185 (23)  | 183 (24)  | 250 (30)  | 275 (30)  | 524 (30)  |
|                  | 546 (30)  | 603 (30)  | 123 (37)  | 422 (37)  | 615 (37)  | 1037 (37) | 1140 (37) | 1871 (37) |
|                  | 62 (39)   | 69 (39)   | 79 (39)   | 81 (39)   | 84 (39)   | 89 (39)   |           |           |
| WriteLn          | 164 ( 8)  | 165 ( 8)  | 271 ( 8)  | 405 (10)  | 406 (10)  | 445 (10)  | 859 (10)  | 956 (10)  |
|                  | 1353 (10) | 1384 (10) | 1533 (10) | 1633 (10) | 1799 (10) | 1888 (10) | 1914 (10) | 88 (12)   |
|                  | 114 (12)  | 126 (12)  | 434 (12)  | 443 (12)  | 603 (12)  | 704 (14)  | 113 (16)  | 775 (16)  |
|                  | 1367 (16) | 1733 (16) | 1983 (16) | 2332 (16) | 2610 (16) | 2623 (16) | 2625 (16) | 2631 (16) |
|                  | 2668 (16) | 56 (17)   | 57 (17)   | 63 (17)   | 64 (17)   | 481 (17)  | 642 (17)  | 649 (17)  |
|                  | 816 (17)  | 1187 (17) | 1188 (17) | 1189 (17) | 1307 (17) | 338 (28)  | 340 (28)  | 344 (28)  |
|                  | 346 (28)  | 348 (28)  | 372 (28)  | 908 (28)  | 1158 (28) | 1159 (28) | 83 (30)   | 84 (30)   |
|                  | 79 (33)   | 186 (33)  | 206 (32)  | 308 (32)  | 323 (32)  |           |           |           |
| WritePt          | 452*(25)  | 1172*(26) | 1185 (26) | 1223 (26) | 1225 (26) | 1311 (40) |           |           |
| WritePtr         | 303 ( 8)  | 305 ( 8)  | 2609 (16) | 455*(25)  | 1194*(26) | 1211 (26) |           |           |
| WriteRect        | 117 (23)  | 458*(25)  | 1220*(26) | 1235 (26) | 1870 (37) | 61 (39)   | 83 (39)   | 84 (39)   |
|                  | 88 (39)   | 89 (39)   | 1312 (40) | 1313 (40) |           |           |           |           |
| WriteRes         | 79*( 5)   | 153*( 5)  | 183*( 5)  | 223*( 5)  | 266*( 5)  | 318*( 5)  | 364*( 5)  | 409*( 5)  |
|                  | 458*( 5)  | 525*( 5)  | 579*( 5)  | 655*( 5)  | 367*( 6)  | 844*( 6)  | 924*( 6)  | 1051*( 6) |
|                  | 1200*( 6) | 1509*( 6) | 1728*( 6) | 1928*( 6) | 2165*( 6) | 2757*( 6) | 2995*( 6) | 3464*( 6) |
|                  | 202*( 9)  | 460*( 9)  | 542*( 9)  | 277*(10)  | 2302*(10) | 2472*(10) | 1486*(13) | 1782*(13) |
|                  | 1942*(13) | 2084*(13) | 2163*(13) | 2243*(13) | 2297*(13) | 275*(18)  | 214*(19)  | 111*(20)  |
|                  | 204*(21)  | 579*(21)  | 1017*(21) | 1285*(21) | 167*(38)  | 164*(40)  |           |           |
| WriteReserves    | 349*(27)  | 1155*(28) |           |           |           |           |           |           |
| WriteSig         | 488*(25)  | 1316*(26) | 1332 (26) |           |           |           |           |           |
| WriteToDeskScrap | 1667*(13) | 2574*(13) | 227 (16)  | 1831*(18) | 309*(23)  | 262*(38)  | 1598*(40) |           |
| WriteVpt         | 464*(25)  | 1270*(26) | 1283 (26) | 1295 (26) | 1297 (26) | 1868 (37) | 652 (40)  | 665 (40)  |
| WriteVRect       | 467*(25)  | 1292*(26) | 1307 (26) |           |           |           |           |           |
| WrLblBoolean     | 156 (15)  | 479*(25)  | 1257*(26) |           |           |           |           |           |
| WrLblField       | 115*(31)  | 255*(32)  | *130 (39) | 147 (39)  |           |           |           |           |
| WrLblHexInt      | 494*(25)  | 1353*(26) |           |           |           |           |           |           |
| WrLblHexLongint  | 498*(25)  | 1376*(26) |           |           |           |           |           |           |
| WrLblPt          | 470*(25)  | 1182*(26) | 1326 (40) |           |           |           |           |           |
| WrLblPtr         | 125 (12)  | 442 (12)  | 473*(25)  | 1208*(26) | 1158 (28) | 1159 (28) |           |           |
| WrLblRect        | 1146 (16) | 3083 (16) | 18 (18)   | 20 (18)   | 22 (18)   | 495 (20)  | 476*(25)  | 1232*(26) |
|                  | 1325 (40) |           |           |           |           |           |           |           |
| WrLblSig         | 147 (15)  | 149 (15)  | 150 (15)  | 490*(25)  | 1330*(26) |           |           |           |
| WrLblVpt         | 152 (15)  | 153 (15)  | 16 (18)   | 497 (20)  | 482*(25)  | 1280*(26) |           |           |
| WrLblVRect       | 485*(25)  | 1304*(26) |           |           |           |           |           |           |
| wSize            | 838*(14)  | 879-(14)  | 881 (14)  | 884 (14)  | 884 (14)  | 904*(14)  | 956-(14)  | 958 (14)  |
|                  | 961 (14)  | 961 (14)  |           |           |           |           |           |           |
| WStateData       | 869 (19)  |           |           |           |           |           |           |           |
| wTopLeft         | 948*(19)  | 954-(19)  | 957 (19)  | 986 (19)  |           |           |           |           |
| WWActivateEvent  | 1312 (16) | 1683 (16) |           |           |           |           |           |           |
| WWAddText        | 866 (41)  | 868 (41)  |           |           |           |           |           |           |
| WWEndForce       | 252 ( 8)  | 699 (26)  | 718 (26)  |           |           |           |           |           |
| WWForceOutput    | 250 ( 8)  | 696 (26)  | 715 (26)  |           |           |           |           |           |
| WWInit           | 442 (14)  | 444 (14)  |           |           |           |           |           |           |
| WWMouseDown      | 1587 (16) |           |           |           |           |           |           |           |
| WWNew            | 519 (14)  | 527 (14)  |           |           |           |           |           |           |
| WWRedirect       | 103 (14)  |           |           |           |           |           |           |           |
| WWUpdateEvent    | 1713 (16) |           |           |           |           |           |           |           |
| -X-              |           |           |           |           |           |           |           |           |
| x                | 82*(12)   | 107*(12)  | 154*(12)  | 163-(12)  | 166 (12)  | 171-(12)  | 171 (12)  | 176*(14)  |
|                  | 198-(14)  | 208-(14)  | 215-(14)  | 1071*(14) | 1099-(14) | 1100 (14) | 1104 (14) | 1105 (14) |
|                  | 278*(25)  | 336*(26)  | 346-(26)  | 347 (26)  | 348 (26)  | 349 (26)  | 330*(28)  | 351 (28)  |
|                  | 352 (28)  | 353 (28)  | 358 (28)  | 359 (28)  | 360 (28)  | 156*(32)  | 163-(32)  | 164 (32)  |

|              |           |           |           |          |         |
|--------------|-----------|-----------|-----------|----------|---------|
|              | 175 (32)  | 192* (32) | 197- (32) | 40* (39) | 41 (39) |
| XFrameOval   | 404* (18) | 488 (18)  |           |          |         |
| XFrameRRect  | 412* (18) | 491 (18)  |           |          |         |
| XLBusyDelay  | 51* (4)   | 209 (4)   |           |          |         |
| XLBusyImage  | 52* (4)   | 128 (4)   |           |          |         |
| XLCursorInit | 53* (4)   | 79 (4)    | 177 (4)   |          |         |
| xLoc         | 354* (37) | 371- (37) | 372 (37)  |          |         |
| XORgn        | 1592 (18) |           |           |          |         |

-Y-

|             |           |          |           |          |  |
|-------------|-----------|----------|-----------|----------|--|
| y           | 278* (25) |          |           |          |  |
| YellowBit   | 72* (6)   | 100 (6)  | 293* (26) | 321 (26) |  |
| yellowColor | 134 (6)   | 805 (26) |           |          |  |

-Z-

|             |            |           |            |  |  |
|-------------|------------|-----------|------------|--|--|
| ZeroScrap   | 225 (16)   | 394 (16)  |            |  |  |
| Zoom        | 1969* (13) | 1632 (16) | 1002* (19) |  |  |
| zoomDocProc | 1270 (16)  |           |            |  |  |
| zoomWindow  | 1011 (19)  |           |            |  |  |
| ZPAOCs      | 37 (32)    | 38* (32)  |            |  |  |
| ZPPAOCs     | 37* (32)   | 59 (32)   |            |  |  |

\*\*\* End PasRef: 4200 id's 41540 references